digital

pdp11

UNIBUS™ interface manual

2nd edition

DEC-11-HIAB-D

PDP-11
INTERFACE MANUAL
SECOND EDITION

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

1st Edition April 1970
2nd Printing (rev) January 1971

Copyright © 1970, 1971 by Digital Equipment Corporation

The material in this manual is for information purposes and is subject to change without notice.

The following are trademarks of Digital Equipment Corporation, Maynard, Massachusetts:

DEC	PDP
FLIP CHIP	FOCAL
DIGITAL	COMPUTER LAB
UNIBUS	

CONTENTS

		Page
CHAPTER 1	**GENERAL DESCRIPTION**	
1.1	Scope	1-1
1.2	Applicable Documents	1-1
1.3	Introduction	1-1
1.3.1	Single Bus	1-1
1.3.2	Bidirectional Lines	1-2
1.3.3	Master-Slave Relation	1-2
1.3.4	Interlocked Communication	1-2
1.4	Peripheral Device Organization and Control	1-3
1.5	Transfer of Bus Master	1-3
1.5.1	Transfer Request Handling	1-3
1.5.2	Priority Structure	1-3
1.5.3	Data Transfer	1-4
1.5.4	Interrupt Requests	1-4
1.5.5	Interrupt Procedure	1-4
CHAPTER 2	**UNIBUS THEORY AND OPERATION**	
2.1	Introduction	2-1
2.2	Unibus Signal Lines	2-1
2.2.1	Data Transfer Lines	2-3
2.2.1.1	Data Lines	2-4
2.2.1.2	Address Lines	2-4
2.2.1.3	Control Signals	2-4
2.2.2	Priority Transfer Lines	2-5
2.2.3	Miscellaneous Control Lines	2-5
2.3	Data Transfer Bus Transactions	2-6
2.3.1	Data Transfer Timing	2-7
2.3.2	DATI and DATIP Bus Transactions	2-8
2.3.3	DATO and DATOB Bus Transactions	2-9
2.4	Priority Transfer Transactions	2-10
2.4.1	Priority Transfer (PTR) Transaction	2-10
2.4.2	Interrupt Transaction	2-12
2.4.3	Priority Chaining	2-12
2.5	Unibus Timing	2-13
2.5.1	Timing Example	2-14
2.5.2	Time-Out Protection	2-14
2.6	Address Mapping	2-17
2.6.1	Interrupt and Trap Vector Locations	2-17
2.6.2	Memory Locations	2-17
2.6.3	Device Register Locations	2-17
2.6.4	Processor Locations	2-19
2.7	Device Registers	2-19

CONTENTS (Cont)

		Page
CHAPTER 3	**INTERFACE CIRCUITS AND HARDWARE**	
3.1	Introduction	3-1
3.2	Circuits	3-1
3.2.1	Unibus Transmission	3-1
3.2.2	Unibus Signal Levels	3-1
3.2.3	Unibus Length and Loading	3-3
3.2.4	Bus Receiver and Transmitter Circuits	3-3
3.3	Unibus Interface Modules	3-4
3.3.1	Unibus Cables	3-5
3.3.2	Unibus Terminations	3-6
3.3.3	Unibus Receivers and Transmitters	3-7
3.3.3.1	M783 Unibus Transmitter	3-7
3.3.3.2	M784 Unibus Receiver	3-8
3.3.3.3	M785 Unibus Transceiver	3-9
3.3.3.4	M798 Unibus Drivers	3-10
3.3.4	M105 Address Selector Module	3-12
3.3.4.1	Inputs	3-12
3.3.4.2	Slave Sync (SSYN)	3-13
3.3.4.3	Outputs	3-13
3.3.4.4	Specifications	3-14
3.3.5	M782 Interrupt Control Module	3-17
3.3.6	M795 Word Count and Bus Address Module	3-23
3.3.7	M796 Unibus Master Control Module	3-25
3.3.8	DR11-A General Device Interface	3-36
3.4	PDP-11 Interface Hardware	3-43
3.4.1	BB11 Blank Mounting Panel	3-43
3.4.2	DD11-A Peripheral Mounting Panel	3-45
3.4.2.1	Unibus	3-45
3.4.2.2	Power	3-45
3.4.2.3	Extended Usage	3-47
3.4.2.4	Grant Continuity	3-47
3.4.2.5	External Device Cables	3-47
3.4.3	BA11 Mounting Boxes	3-47
3.4.4	H720 Power Supply	3-49
CHAPTER 4	**INTERFACE EXAMPLES**	
4.1	Introduction	4-1
4.2	Basic Interface	4-1
4.2.1	Interface Operation	4-1
4.2.2	Data Transfer Operation	4-1
4.2.3	Circuit Implementation	4-2
4.2.4	Programming the Interface	4-5
4.3	Programmed Device Interface	4-5
4.3.1	Analog-to-Digital Converter	4-5

CONTENTS (Cont)

		Page
4.3.2	Interface Description	4-7
4.3.3	Transfer Operations	4-7
4.3.4	Circuit Implementation	4-7
4.3.5	Programming the Interface	4-7
4.4	Interrupt Serviced Interface	4-8
4.4.1	Interface Description	4-11
4.4.2	Circuit Implementation	4-11
4.4.3	DR11-A Implementation	4-11
4.4.4	Interface Programming	4-16
4.5	Direct Memory Access (DMA) Interface	4-17
4.5.1	Interface Description	4-17
4.5.2	Interface Implementation	4-17
4.5.3	Programming the Interface	4-18
4.5.4	Interface Operation Timing	4-18
4.5.5	Interface Options	4-18
4.6	Output Interface with Interrupt Control	4-23
4.6.1	Device Description	4-23
4.6.2	Interface Description	4-23
4.6.3	Interface Operation	4-23
4.6.4	Circuit Implementation	4-23
4.6.5	Interface Programming	4-24
4.7	DAC-DMA Interface	4-27
4.7.1	Interface Description	4-27
4.7.2	Interface Operation	4-29
4.7.3	Interface Options	4-29
4.8	PDP-11 to Data Channel Interface	4-33
4.8.1	Data Channel Description	4-33
4.8.2	Interface Description	4-34
4.8.3	Interface Operation	4-35
4.8.4	Interface Implementation	4-35
4.9	PDP-11 to PDP-11 Interface	4-35
4.9.1	Interface Description and Operation	4-35
4.9.2	Interface Implementation	4-38
4.9.3	Interface Programming	4-38
4.10	Memory Increment Interface	4-40
4.10.1	Interface Description and Operation	4-40
4.10.2	Optional Arrangements	4-42

APPENDICES

		Page
APPENDIX A	**PROCESSOR BLOCK DIAGRAM**	
A.1	Introduction	A-1
A.2	Block Diagram	A-1
A.2.1	Unibus Interface	A-1
A.2.2	Data Paths	A-1
A.2.3	General-Purpose Registers	A-1
A.3	Major States	A-3
A.4	Bus Cycles	A-4
APPENDIX B	**INSTRUCTION SET**	
B.1	Introduction	B-1
B.2	Instruction Groups	B-1
B.3	Major States	B-1
B.3.1	Fetch	B-2
B.3.2	Source and Destination	B-2
B.3.3	Execute	B-2
B.3.4	Service	B-5
APPENDIX C	**UNIBUS PIN ASSIGNMENTS**	C-1

ILLUSTRATIONS

Figure No.	Title	Art No.	Page
1-1	PDP-11 System-Simplified Block Diagram	11-0034	1-2
2-1	PDP-11 Interface Block Diagram	11-0017	2-1
2-2	Data Line Bit Format	11-0033	2-4
2-3	Address Line Bit Format	11-0033	2-4
2-4	DATI and DATIP Timing Diagram	11-0002	2-8
2-5	DATO or DATOB Timing Diagram	11-0003	2-9
2-6	PTR Timing Diagram	11-0004	2-11
2-7	INTR Timing Diagram	11-0005	2-12
2-8	Priority Chaining Example	11-0006	2-13
2-9	Typical DATO Timing Flow	11-0181	2-15
2-10	Slave Logic for DATO (Fast Response)	11-0181	2-15
2-11	Typical DATI Timing Flow	11-0182	2-16
2-12	Slave Logic for DATI (Fast Response)	11-0182	2-16
2-13	PDP-11 Address Map	11-0191	2-18
2-14	Preferred CSR Bit Assignments	11-0016	2-19
3-1	Bus Terminations for Bidirectional (a.) and Unidirectional (b.) Bus Lines	11-0013	3-2
3-2	Transmitter and Receiver Equivalent Circuits	11-0031	3-3
3-3	Transmitter and Receiver Typical Circuits	11-0030	3-5

ILLUSTRATIONS (Cont)

Figure No.	Title	Art No.	Page
3-4	Unibus Jumper Module M920	4738	3-5
3-5	Unibus Cable BC11A	4959-1	3-6
3-6	M783 Unibus Transmitter (schematic diagram)	11-0023	3-7
3-7	M784 Unibus Receiver (schematic diagram)	11-0022	3-8
3-8	M785 Unibus Transceiver (schematic diagram)	11-0024	3-9
3-9a	One of 16 Circuits on the M798	11-0174	3-10
3-9b	Typical Use of M798	11-0175	3-10
3-10	M798 Unibus Drivers (schematic diagram)	11-0172	3-11
3-11	M105 Address Selector	11-0046	3-12
3-12	Device Register Select Address Format	11-0029	3-13
3-13	M105 Address Selector (schematic diagram)	11-0020	3-15
3-14	M782 Interrupt Control (block diagram)	11-0028	3-17
3-15	M782 Interconnection for 2-Channel Interrupt	11-0027	3-18
3-16	M782 Interconnection for Direct Memory Access	11-0047	3-19
3-17	State Diagram of Master Control	11-0026	3-19
3-18	M782 Interrupt Control (schematic diagram)	11-0019	3-21
3-19	M795 Word Count and Bus Address (block diagram)	11-0185	3-24
3-20	M795 Word Count and Bus Address (schematic diagram)	11-0168	3-27
3-21	M796 Unibus Master Control (block diagram)	11-0176	3-29
3-22	M796 Unibus Master Control (typical use)	11-0171	3-31
3-23	M796 Timing Diagram for DATO	11-0173	3-32
3-24	M796 Timing Diagram for DATI	11-0184	3-32
3-25	M796 Unibus Master Control (schematic diagram)	11-0169	3-33
3-26	DR11-A Device Register Interface	5221-2	3-37
3-27	DR11-A Device Register Interface (block diagram)	11-0021	3-38
3-28	DR11-A Address and Bit Assignments	11-0025	3-40
3-29	DR11-A Physical Layout	11-0178	3-40
3-30	M786 General Device Register (schematic diagram)	11-0167	3-41
3-31	Basic PDP-11 Layout with Two BB11 Panels	11-0007	3-44
3-32	BB11 System Unit	4959-2	3-44
3-33	BB11 Module Layout	11-0035	3-45
3-34	DD11-A Module Layout	11-0192	3-46
3-35	Device Control as Wired in the DD11-A	11-0195	3-46
3-36	BA11 Mounting Box	4959-3	3-48
4-1	Basic Interface (block diagram)	11-0180	4-2
4-2	Basic Interface (schematic diagram)	11-0074	4-3
4-3	Programmed Device Interface (block diagram)	11-0194	4-6
4-4	Programmed Device Interface (schematic diagram)	11-0188	4-9
4-5	Interrupt Serviced Interface (block diagram)	11-0193	4-12
4-6	Interrupt Serviced Interface (schematic diagram)	11-0038	4-13
4-7	DR11-A Implementation (block diagram)	11-0021	4-15
4-8	DMA Interface (block diagram)	11-0044	4-19
4-9	DMA Interface (timing diagram)	11-0045	4-20
4-10	DMA Interface (schematic diagram)	11-0177	4-21

ILLUSTRATIONS (Cont)

Figure No.	Title	Art No.	Page
4-11	Output Interface with Interrupt Control (block diagram)	11-0051	4-24
4-12	Output Interface with Interrupt Control (schematic diagram)	11-0073	4-25
4-13	DAC-DMA Interface (block diagram)	11-0179	4-28
4-14	DAC-DMA Interface (schematic diagram)	11-0183	4-31
4-15	DAC-DMA Interface (timing diagram)	11-0189	4-33
4-16	PDP-11-to-Data Channel Interface (block diagram)	11-0057	4-34
4-17	PDP-11-to-Data Channel Interface (control circuit schematic)	11-0056	4-36
4-18	PDP-11-to-PDP-11 Interface (block diagram)	11-0059	4-37
4-19	PDP-11-to-PDP-11 Interface Control (circuit schematic)	11-0058	4-39
4-20	Memory Increment Interface (block diagram)	11-0062	4-41
4-21	Memory Increment Interface (timing diagram)	11-0190	4-42
4-22	Memory Increment Interface (control circuit schematic)	11-0170	4-43
A-1	PDP-11 Processor Block Diagram	11-0010	A-2
A-2	Data Flow	11-0011	A-2

TABLES

Table No.	Title	Page
2-1	Unibus Signals	2-2
2-2	Data Transfer Signals	2-3
2-3	Data Transfer Operations	2-5
2-4	Bus-Data Transfer Transactions	2-6
2-5	Preferred Order of Device Register Assignments	2-20
3-1	Unibus Receiver and Transmitter Characteristics	3-4
3-2	M105 Select Lines	3-13
3-3	Gating Control Signals	3-13
3-4	Summary of M782 Signals	3-23
3-5	M795 Input Signals	3-24
3-6	M795 Output Signals	3-25
3-7	Control Line Input States for M796	3-30
3-8	M796 Input Signals	3-35
3-9	M796 Output Signals	3-35
3-10	DR11-A Pin Assignments	3-39
3-11	BA11 Mounting Box Models	3-49
3-12	H720 Power Supply Outputs	3-49
4-1	Bus Line Gating	4-38
A-1	Processor Major States	A-3
A-2	Typical Major State Sequences	A-4

TABLES (Cont)

Table No.	Title	Page
B-1	Address Modes (Source or Destination)	B-2
B-2	Double-Operand Group	B-3
B-3	Single-Operand Group	B-3
B-4	Operate Group	B-4
B-5	Subroutine Group	B-4
B-6	Condition Code Operators	B-4
B-7	Conditional Branches	B-5
B-8	Bus Cycles During Service	B-5
C-1	Unibus Pin Assignments (by pin numbers)	C-1
C-2	Unibus Pin Assignments (by Signal name)	C-2
C-3	BB11 Power Pin Assignments	C-3

CHAPTER 1
GENERAL DESCRIPTION

1.1 SCOPE

This manual provides detailed interfacing information for PDP-11 System users. It is assumed that the reader is familiar with the PDP-11 System and has a thorough understanding of the principles and concepts introduced in the *PDP-11 Handbook*. This handbook presents a comprehensive overview of the system, a detailed description of the instruction set, and details of peripheral programming.

This manual describes detailed operation of the Unibus and methods for interfacing to the Unibus™ for the addition of custom-designed peripheral equipment to the PDP-11 System. It is beyond the scope of this manual to provide detailed theory of operation for the processor or the instruction set; however, a short description of the processor is presented in Appendix A, and a description of bus cycles relating to the instruction set is presented in Appendix B.

1.2 APPLICABLE DOCUMENTS

The following documents will help the reader understand interface techniques and the overall PDP-11 System:

a. *PDP-11 Handbook*

b. *KA11* or *KC11 Processor Manual*

c. *Digital 1970 Logic Handbook*

The *PDP-11 Handbook* introduces the overall system and describes the instruction set. The *PDP-11 Unibus Interface Manual* and the *KA11* or *KC11 Processor Manual*, used together, completely describe the PDP-11/15 and PDP-11/20 Systems. The primary subject of the *PDP-11 Interface Manual* is the Unibus; the primary subject of the *KA11* or *KC11 Manual* is the processor. Many logic circuits used to implement the interface examples are covered in the *Digital 1970 Logic Handbook*.

1.3 INTRODUCTION

Digital Equipment Corporation's PDP-11 is a 16-bit, general-purpose, parallel-logic computer that uses 2's complement arithmetic. All communication between system components is accomplished by a single high-speed bus called the Unibus. Four concepts are extremely important for an understanding of the hardware and software implications of the Unibus. Each concept is covered separately in subsequent paragraphs.

1.3.1 Single Bus

The Unibus is a single, common path that connects the processor, memory, and all peripherals. Addresses, data, and control information are transmitted along the 56 lines of the bus. Figure 1-1 is a simplified block diagram of the PDP-11 System and Unibus.

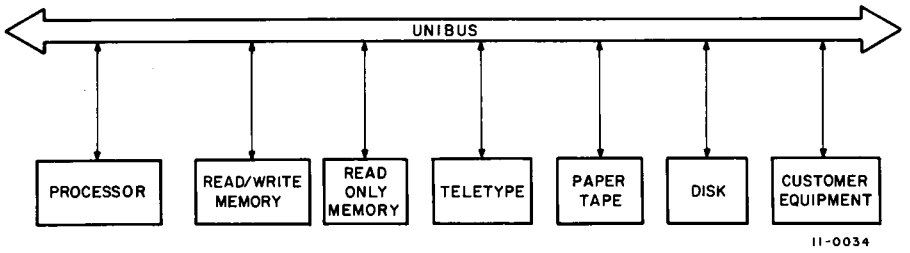


Figure 1-1 PDP-11 System-Simplified Block Diagram

The form of communication is the same for every device on the Unibus. The processor uses the same set of signals to communicate with memory and peripheral devices. Peripheral devices also use this set of signals when communicating with the processor, memory, or other peripheral devices.

All instructions applied to data in memory can be applied equally well to data in peripheral device registers. Therefore, peripheral device registers may be manipulated as flexibly as memory by the processor. This is an especially powerful feature, considering the special capability of PDP-11 instructions to process data in any memory location as though it were an accumulator.

1.3.2 Bidirectional Lines

Most Unibus lines are bidirectional; therefore, the input lines can also be driven as output lines. This means that a peripheral device register can be either read or changed by the processor or other peripheral devices, and the same data lines can be used for transfer operations. Thus, the same register can be used for both input and output functions.

1.3.3 Master-Slave Relation

Communication between two devices on the bus is in a master-slave relationship. During any bus operation, one device has control of the bus. This device, the bus master, controls the bus when communicating with another device on the bus, called the slave. A typical example of this relationship is the processor, as master, fetching an instruction from memory (which is always a slave). Another example is the disk, as master, transferring data to memory, as slave. Master-slave relationships are dynamic. The processor, for example, passes bus control to a disk. The disk, as master, then communicates with a slave memory bank.

The Unibus is used by the processor and all I/O devices; thus, a priority structure determines which device obtains control of the bus. Consequently, every device on the Unibus capable of becoming bus master has an assigned priority. When two devices, which are capable of becoming bus master, have identical priority values and simultaneously request use of the bus, the device that is electrically closest to the bus receives control.

1.3.4 Interlocked Communication

Communication on the Unibus is interlocked between devices. Each control signal issued by the master device must be acknowledged by a response from the slave to complete the transfer. Therefore, communication is independent of the physical bus length and the response time of the master and slave devices. The maximum transfer rate on the Unibus with optimum device design, is one 16-bit word every 400 ns, or 2.5 million 16-bit words per second.

1.4 PERIPHERAL DEVICE ORGANIZATION AND CONTROL

Registers in peripheral devices are assigned addresses similar to memory; thus, all PDP-11 instructions that address memory locations can become I/O instructions. Data registers in devices can take advantage of all the arithmetic power of the processor. The PDP-11 controls devices differently than most computer systems. Control functions are assigned a register address, and then the individual bits within that register can cause control operations to occur. For example, the command to make the paper-tape reader read a frame of tape is provided by setting a bit (the reader enable bit) in the control register of the device. Instructions such as MOV and BIS may be used for this purpose. Status conditions are also handled by the assignment of bits within this register, and the status is checked with TST, BIT, and CMP instructions. In addition, there is no limit to the number of registers that a device may have, providing an unlimited flexibility in the design and control of peripheral equipment.

1.5 TRANSFER OF BUS MASTER

A device (other than the processor) that is capable of becoming bus master generally requests use of the bus for one of two purposes:

a. to make a non-processor transfer of data directly to or from memory, or

b. to interrupt program execution and force the processor to branch to a specific address where an interrupt service routine is located.

Subsequent paragraphs discuss priority structure and the two main uses of the bus: non-processor transfers and interrupts.

1.5.1 Transfer Request Handling

The request and granting of bus mastership is performed in parallel with data transfers on a completely independent set of bus lines. Thus, while one device is using the bus, the next request is being checked for priority and the next user is being assigned. Because of this time parallelism, successive data transfers by different master devices can occur at the full Unibus speed.

1.5.2 Priority Structure

When a device capable of becoming bus master requests use of the bus, the handling of that request depends on the location of that device in the priority structure. The following factors must be considered to determine the priority of the request:

a. The processor's priority is set under program control to one of eight levels using bits 7, 6, and 5 in the processor's status register. These three bits set a priority level that inhibits granting of bus requests on the same or lower levels.

b. Bus requests from external devices can be made on any one of five request lines. A non-processor request (NPR) has the highest priority, and its request is granted by the processor between bus cycles of an instruction execution. Bus request 7 (BR7) is the next highest priority and bus request 4 (BR4) is the lowest. The four lower level priority requests (BR7 to BR4) are granted by the processor between instructions. When the processor priority is set to a specific level, all bus requests on that level and below are ignored. For example, if the processor priority is 6, requests on BR6 or any other lower level are not granted.

c. When more than one device is connected to the same bus request line, the device electrically nearer the processor has a higher priority than the device further away. Any number of devices can be connected to a specific BR or NPR line.

When a device other than the processor gains control of the bus, it uses the bus to perform either a data transfer or an interrupt request as described in the following paragraphs.

1.5.3 Data Transfer

Direct memory or device access data transfers can be accomplished between any two peripherals without processor supervision. These are called NPR level data transfers. Normally, NPR transfers are made between the memory and a mass storage device, such as a disk.

During NPR transfers, it is not necessary for the processor to transfer the information between the memory and the mass storage device. The bus structure enables device-to-device transfers, thereby allowing customer-designed peripheral controllers to directly access other devices (such as disks) on the bus. This direct access capability permits operations such as a disk directly refreshing a CRT display.

An NPR device provides extremely fast access to the bus and can transfer data at high rates once it gains control. The processor state is not affected by this type of transfer; therefore, the processor can relinquish bus control while an instruction is in progress. This release of the bus can normally occur at the end of any bus cycle, however, the bus can never be released between cycles of a read-modify-write sequence. (This is described more fully in Chapter 2.) In PDP-11/15 and 11/20 Systems containing a KA11 or KC11 Processor, an NPR device gains bus control in 3.5 μs or less. An NPR device in control of the bus transfers 16-bit words to memory at the same speed as the memory cycle time. In the case of the MM11-E Core Memory, transfer occurs every 1.2 μs, and in the MM11-F Core Memory, every 950 ns. With interleaved memory MM11-EX or MM11-FX, the times are faster, up to one word every 490 ns on transfers to memory.

1.5.4 Interrupt Requests

Devices that gain bus control with one of the bus request lines (BR7, BR6, BR5, BR4) can take full advantage of the power and flexibility of the processor by requesting an interrupt. The entire instruction set is then available for manipulating data and status registers. When a device servicing program is to be run, the task being performed by the processor is interrupted, and the device service routine is initiated. After the device request has been satisfied, the processor returns to its former task. Note that interrupt requests can be made only if bus control has been gained through a BR priority level. An NPR level request can never be used for an interrupt request.

1.5.5 Interrupt Procedure

This paragraph provides an example of an interrupt operation. Assume that a peripheral requires service and requests use of the bus at one of the four BR levels. The operations required to service the device are as follows:

a. Priorities permitting, the processor relinquishes bus control to the device.

b. When the device gains control of the bus, it sends the processor an interrupt command and a unique address of a memory location which contains the starting address of the device service routine. (This is called the interrupt vector address.) Immediately following this pointer address is a word (located at vector address +2) to be used as the new processor status (PS) word.

c. The processor pushes the current processor status word and then the current program counter (PC) value on the processor stack. The stack is pointed to by register R6.

d. The new PC and PS (the interrupt vector) are taken from the address specified by the device, and the device service routine is initiated.

NOTE

These operations are performed automatically and no device polling is required to determine which service routine to execute.

e. These operations are performed in 7.2 μs in the KA11 and KC11 processors from the time the processor receives the interrupt command until it fetches the first instruction of the service routine. Note that this time interval assumes that no NPR transfers occurred during this time.

f. The device service routine can cause the processor to resume the interrupted process by executing the return from interrupt (RTI) instruction which pops the two top words from the processor stack and transfers them back to the PC and PS registers.

g. A device service routine can, in turn, be interrupted by a higher priority bus request any time after the first instruction of the routine has been executed.

h. If such an interrupt occurs, the PC and PS of the current device service routine are automatically pushed onto the stack, and the new device routine is initiated as before. This nesting of priority interrupts can continue to any level; the only limitation is the amount of memory available for the processor stack.

CHAPTER 2
UNIBUS THEORY AND OPERATION

2.1 INTRODUCTION

This chapter provides detailed theory of Unibus operation and includes descriptions of bus signal lines, bus transactions, bus timing, and bus interfacing characteristics. In addition, this chapter contains discussions of address allocations and device registers.

2.2 UNIBUS SIGNAL LINES

The PDP-11 Unibus consists of 56 signal lines. All devices, including the processor, are connected to these lines in parallel (see Figure 2-1). The bidirectional nature of 51 signal lines permits signals to flow in either direction. The remaining five unidirectional lines are used for priority bus control. All 56 signals and their functions are listed in Table 2-1 and are described in subsequent paragraphs. In addition, Unibus pin assignments are listed in Appendix C.

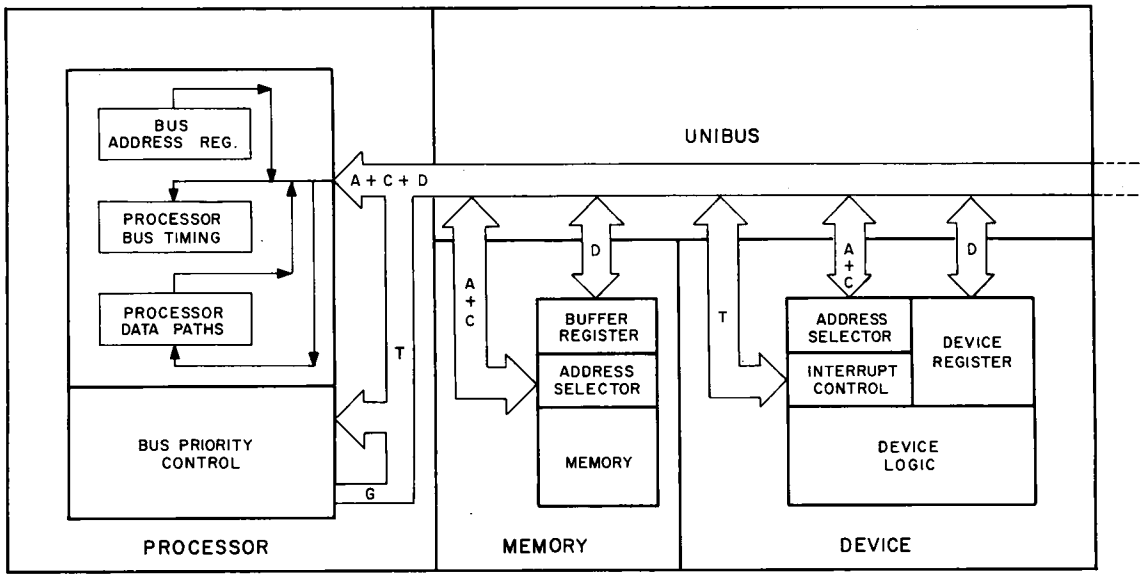


Figure 2-1 PDP-11 Interface Block Diagram

Table 2-1
Unibus Signals

Name	Mnemonic	Source	Destination	Timing	Function
Data Transfer Signals (For transfer of data to or from master)					
Address	A⟨17:00⟩	Master	All	MSYN	Selects slave device
Data	D⟨15:00⟩	Master	Slave	MSYN (DATO, DATOB)	
		Slave	Master	SSYN (DATI, DATIP)	
Control	C⟨1:0⟩	Master	Slave	MSYN	Selects transfer operation
Master Sync	MSYN	Master	Slave	Beginning of transfer	Initiates operation and gates A, C and D signals
Slave Sync	SSYN	Slave	Master	Data accepted (DATO, DATOB) Data Available (DATI, DATIP)	Response to MSYN
Parity Bit Low	PA	Master	Slave	Same as Data	Transmits parity bit, low byte
Parity Bit High	PB	Master	Slave	Same as Data	Transmits parity bit, high byte
Priority Transfer Signals (For transfer of bus control to a priority-selected master)					
Non-processor Request	NPR	Any	Processor	Asynchronous	Highest priority bus request
Bus Request	BR⟨7:4⟩	Any	Processor	Asynchronous	Requests bus mastership
Non-Processor Grant	NPG	Processor	Next master	In parallel with data transfer	Transfers bus control
Bus Grant	BG⟨7:4⟩	Processor	Next master	After instruction	Transfers bus control
Selection Acknowledge	SACK	Next Master	Processor	Response to NPG or BG except during transfer of control	Acknowledges grant & inhibits further grants
Bus Busy	BBSY	Master	All		Asserts bus mastership
Interrupt	INTR	Master	Processor	After asserting BBSY (not after NPR) device may perform several transfers before asserting INTR.	Transfers bus control to handling routine in processor

Table 2-1 (Cont)
Unibus Signals

Name	Mnemonic	Source	Destination	Timing	Function
Miscellaneous Signals					
Initialize	INIT	Processor	All	Asynchronous	Clear and reset signal
AC low	AC LO	Power	All	Asynchronous	Indicates impending power failure
DC low	DC LO				Indicates dc voltages out of tolerance, and system operation must be suspended.

NOTE
Signals on the Unibus are asserted when *low* (except for the unidirectional bus grant lines). All timing diagrams in this manual reflect the asserted and cleared levels of Unibus lines as described in Paragraph 3.2.2.

2.2.1 Data Transfer Lines

Forty bidirectional bus lines are used for data transfer. In a data transfer, one device is bus master and controls the transfer of data to or from a slave device. The processor is always bus master when no other device is using the bus, and it is master for all data transfers involved in normal instruction processing.

Data transfer signals are listed in Table 2-2

Table 2-2
Data Transfer Signals

Name	Mnemonic	No. of Lines
DATA	D⟨15:00⟩*	16
ADDRESS	A⟨17:00⟩	18
CONTROL	C⟨1:0⟩	2
MASTER SYNC	MSYN	1
SLAVE SYNC	SSYN	1
PARITY BIT LOW	PA	1
PARITY BIT HIGH	PB	1
		TOTAL: 40

*Throughout this manual, the notation A⟨a:b⟩ specifies a-b+1 signal lines, which are named Aa through Ab. Thus, D⟨15:00⟩ specifies 16 data signal lines which are named D15 through D00.

Simplified and standardized control logic is made possible by using separate dedicated lines for all signals. In any data transfer, data is transmitted and received; the master device provides the address of the slave device; and control and timing signals are provided. Each of these three functions occurs on a distinct set of bus lines, eliminating

the use of additional hardware and extra timing states to distinguish between address, control information, and data.

2.2.1.1 **Data Lines (D⟨15:00⟩)** – The 16 data lines are used to transfer information between master and slave. The bit format is shown in Figure 2-2

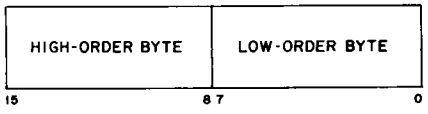


Figure 2-2 Data Line Bit Format

2.2.1.2 **Address Lines (A⟨17:00⟩)** – The 18 address lines are used by the master device to select the slave (a unique memory or device register address) with which it will communicate. The reason for 18 address lines is to extend the total memory capability of future members of the PDP-11 family to 262,144 bytes. The bit format of the 18 signals is shown in Figure 2-3.

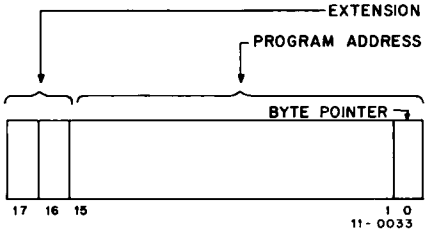


Figure 2-3 Address Line Bit Format

Lines A⟨17:01⟩ specify a unique 16-bit word. In byte operations, A00 specifies the byte being referenced. If a word is referenced at X (X must be even, since words can be addressed on even boundaries only), the low-order byte can be referenced at X and the high-order byte at X+1.

Only 16 bits are normally supplied by programs as memory reference addresses. In the processor used with PDP-11/15 and PDP-11/20 Systems, lines A17 and A16 are asserted (forced to 1) whenever the processor attempts to reference an address between 160000 and 177777; i.e., where A15=A14=A13=1. Thus, the processor converts this 16-bit address to a full 18-bit bus address.

Peripheral devices are normally assigned an address from within the bus address allocations from 760000–777777 (program addresses, 160000–177777).

2.2.1.3 **Control Signals** – The control signals are divided into three groups: signals that select data transfer operations, signals that allow the master and slave device to communicate, and signals used for parity checking.

a. *Control Lines (C⟨1:0⟩)* – These two bus signals are coded by the master device to control the slave in one of four possible data transfer operations shown in Table 2-3.

b. *Master and Slave Synchronization* – Master synchronization (MSYN) is a control signal used by the master to indicate to the slave that address and control information is present. Slave synchronization (SSYN) is the slave's response to the master (usually a response to MSYN).

c. *Parity Bit Low (PA) and Parity Bit High (PB)* – These signals are for devices on the Unibus that use parity checks. PA is the parity of the high-order byte (that transferred on D⟨15:08⟩), and PB is the parity of the low-order byte (D⟨07:00⟩). These lines are used by the MP11 Parity Option in conjunction with parity memories such as the MM11-FP. Future uses of PA and PB for total system (Unibus) parity is also reserved for high security applications.

Table 2-3
Data Transfer Operations

C1	C0	Operation
0	0	DATI-Data In
0	1	DATIP-Data In, Pause
1	0	DATO-Data Out
1	1	DATOB-Data Out, Byte

2.2.2 Priority Transfer Lines

The Unibus contains 13 lines classified as priority transfer lines. Five of these are priority bus request lines (BR⟨7:4⟩, NPR) and five are the corresponding grant lines (BG⟨7:4⟩, NPG) which the processor uses to respond to a specific bus request. Each device of the same priority level passes a grant signal to the next device on the line, unless it has requested bus control; in this case, the requesting device blocks the signal from the following devices and assumes bus control. A discussion of physical chaining of devices to create priority levels is presented in Paragraph 2.4.3. In addition, there are three other control lines: SACK, BBSY, and INTR. All 13 lines are described below.

a. *Bus Request Lines (BR⟨7:4⟩)* – These four bus signals are used by peripheral devices to request control of the bus.

b. *Bus Grant Lines (BG⟨7:4⟩)* – These signals are the processor's response to a bus request. They are asserted only at the end of instruction execution, and in accordance with the priority determination.

c. *Non-Processor Request (NPR)* – This signal is a bus request from a peripheral device to the processor.

d. *Non-Processor Grant (NPG)* – This signal is the processor's response to an NPR. It occurs at the end of a bus cycle.

e. *Selection Acknowledge (SACK)* – SACK is asserted by a bus-requesting device that has received a bus grant. Bus control passes to this device when the current bus master completes its operation. (If SACK is not received within 10 μs, time out occurs and the bus grant is cleared automatically by the processor.)

f. *Interrupt (INTR)* – This signal is asserted by the bus master to start a program interrupt in the processor.

g. *Bus Busy (BBSY)* – This signal is asserted by the master devices to indicate bus is being used.

2.2.3 Miscellaneous Control Lines

There are two additional lines on the Unibus which may be used by all devices. These lines are: initialize and ac line low. In addition, one signal is available for devices built on system units that are mounted within a BA11 Mounting Box and powered by an H720 Power Supply. This signal is dc line low (DC LO) which is also a Unibus signal.

a. *Initialization (INIT)* – This signal is asserted by the processor when the START key on the console is depressed, when a RESET instruction is executed, or when the power fail sequence occurs. In the latter case, INIT is asserted following the power fail service routine while power is going down, and again when power comes up. INIT may also be used to clear and initialize peripheral devices by means of the RESET instruction.

b. *AC Line Low (AC LO)* – This is an anticipatory signal which starts the power fail trap sequence, and may also be used in peripheral devices to terminate operations in preparation for power loss. When AC LO is cleared, the power up instruction sequence in the processor begins. It is the programmer's responsibility to make certain that the trap vector is loaded with a pointer to the power fail routine. If this is not done, an undefined sequence results.

c. *DC Line Low (DC LO)* – This signal, which comes from the H720 Power Supply, is wired from the power connector card slot to the Unibus on all system units. This signal remains cleared as long as all dc voltages are within specified limits. If an out-of-voltage condition occurs, DC LO is asserted by the power supply. Devices such as core memories use the DC LO signal to inhibit further operations. The DC LO signal is normally cleared before AC LO when power is coming up and is asserted after AC LO when power is going down. Note that the power fail trap is initiated by AC LO only and that the DC LO signal is used by the processor to cause INIT on the bus.

2.3 DATA TRANSFER BUS TRANSACTIONS

All bus activity is asynchronous and depends on interlocking of control signals. In every case, a signal from a slave device is generated in response to a signal from a master device, and the master signal is dropped in response to the slave signal. The complete elimination of critical self-timing gives the bus the flexibility to operate with devices running at different speeds.

NOTE

Signals on the Unibus are asserted when *low* (except for the unidirectional bus grant lines). All timing diagrams in this manual reflect the asserted and cleared levels of Unibus lines as described in Paragraph 3.2.2.

The four bus-data transfers are described in Table 2-4. The bus master determines one of the four data transfers by asserting the proper code on the C⟨1:0⟩ lines.

NOTE

All data transfers are with reference to the master device; data-in is always from slave to master, and data-out is from master to slave. For example, when the processor (master) loads data into memory (slave), a data-out bus operation is performed.

Table 2-4
Bus-Data Transfer Transactions

Name	Mnemonic	C Lines C1	C Lines C0	Function	Octal Code
Data in	DATI	0	0	Data from slave to master	0
Data in, pause	DATIP	0	1	Inhibits restore cycle in destructive read-out devices; Pause flip-flop is set which inhibits clear	1

Table 2-4 (Cont)
Bus-Data Transfer Transactions

Name	Mnemonic	C Lines C1	C Lines C0	Function	Octal Code
Data in, pause (cont)				cycle on following DATO (B). Must be followed by DATO or DATOB.	
Data out	DATO	1	0	Data from master to slave.	2
Data out, Byte	DATOB	1	1	Transfers data from master to a single byte in slave. Data transmitted on D⟨15:08⟩ for A00=1 D⟨07:00⟩ for A00=0.	3

The DATI and DATIP transactions request transfer of data from a slave, the address of which is specified by A⟨17:00⟩, to the master. Both transfers use the data lines to carry the data. There is no distinction made by the slave as to whether the transfer is used for byte or word data. The slave places the data on D⟨15:00⟩. It is the function of the master device to retrieve the data from the proper lines: low-order byte register with A00=0 from D⟨07:00⟩; high-order byte register with AA00=1 from D⟨15:08⟩; or word register from D⟨15:00⟩. The DATIP operation is identical to the DATI, except DATIP is used to inform the slave device that this is the first part of an in-modify-out cycle. A DATIP normally sets a pause flag in the destructive read-out device (i.e., memory) which inhibits the restore cycle. The DATIP must be followed by a data-out cycle (DATO or DATOB), and the master must retain bus control until it is completed. In nondestructive readout devices (i.e., flip-flops), the DATI and DATIP are treated identically.

The DATO and DATOB operations transfer data from the master to the slave. A DATO is used to transfer a word to the address specified by A⟨17:01⟩. The slave ignores A00 and the master places data on D⟨15:00⟩. A DATOB is used to transfer a byte of data to the address specified by A⟨17:00⟩. Line A00=0 indicates the low-order byte, and the master places the data on lines D⟨07:00⟩.

Appendix B lists all the bus cycles used by the KA11 and KC11 processors during the execution of instructions.

2.3.1 Data Transfer Timing

The design of the Unibus imposes certain timing restrictions although transfers are interlocked. Responsibility for these timing restrictions has been assigned to the master to simplify the slave design.

In all transfers, it is assumed that there can be a maximum 75-ns skew due to driver, receiver, and transmission line tolerances. In other words, the coincident assertion of two lines at the transmitter inputs of one device could result in a maximum difference of 75-ns in the occurrence of those signals at the receiver outputs in another device.

Because of this possible skew, the master always delays its MSYN signal to ensure that MSYN does not reach the slave device prior to valid data or addresses. In addition, the MSYN signal is further delayed to allow 75 ns for decoding by the slave device. The master also must not drop the A (address) or C (control) lines until 75 ns after MSYN has been dropped to guarantee that there are no spurious selections. Note, however, that when a slave transmits data to a master (DATI or DATIP), the deskew and decode time delay must be made by the master (refer to Paragraph 2.3.2 f). Additional timing information can be found in Paragraph 2.5.

2.3.2 DATI and DATIP Bus Transactions (See Figure 2-4)

All data transfer functions are with reference to the master device; therefore, data-in (DATI) indicates data transfer from the slave to the master.

a. The master device places the address (or location) of the slave device on the A lines, and places 0 on the C lines for a DATI, or 1 for a DATIP. The master device then waits 150 ns (minimum): 75 ns to allow for worst-case signal skew, and 75 ns to allow internal logic in the slave devices to decode the address.

b. The slave device decodes the address and control bits to determine if the slave is to participate in a data transfer.

c. If the slave device from a previous bus transaction still has SSYN asserted, the master device waits until SSYN is clear. MSYN is asserted when SSYN is clear and the delay from step *a* is complete.

d. When the MSYN signal is received by the slave device, the device prepares the data for transmission to the master. For devices such as memory, this means performing a read cycle. For flip-flop registers, the data is available immediately.

e. When data is available, the slave places data on the D lines and asserts SSYN. If the slave is a destructive read-out device, it enters a restore cycle, if the command was a DATI; for a DATIP, the slave sets a pause flag and waits for modified data before performing a write cycle.

f. The master device receives SSYN and the data. After a 75-ns minimum delay to allow for skew and time for internal gating, the master device strobes the data and clears MSYN. If the processor is master and SSYN is not received within 20 μs, time-out occurs, the MSYN signal is cleared automatically by the processor, and an error trap occurs.

g. After a 75-ns minimum wait, to ensure that no spurious device selections occur as various bits of the address change while MSYN is still asserted, the A and C lines are cleared.

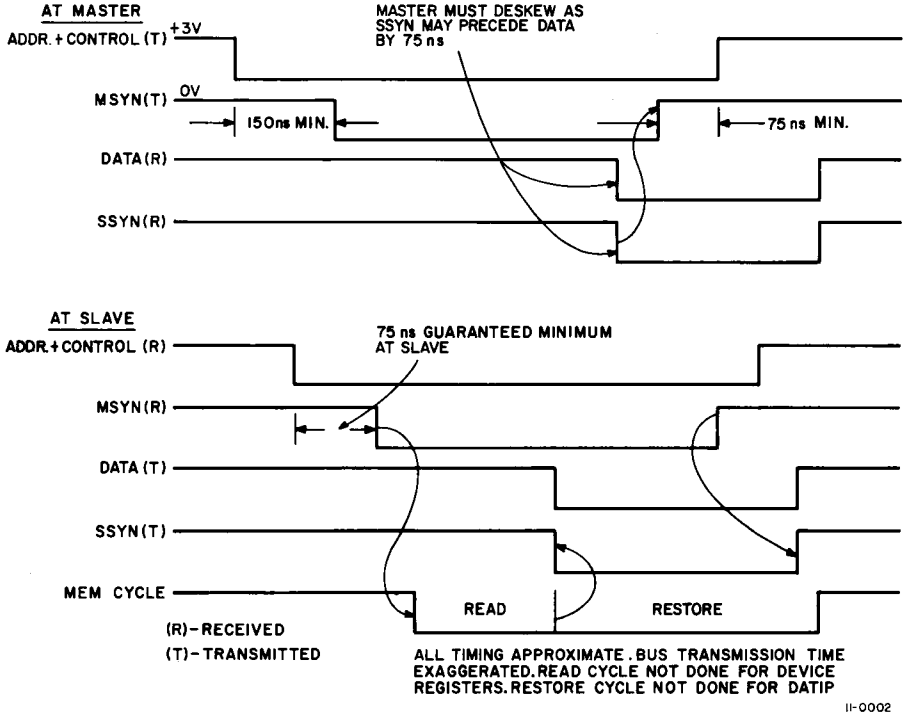


Figure 2-4 DATI and DATIP Timing Diagram

h. When MSYN is cleared, the slave clears the D lines and SSYN; the bus is now free for other use.

i. The master receives the cleared SSYN, which signals the end of the current bus transaction. If an output transfer follows (a DATO or a DATOB must follow a DATIP), step *a* of the DATO(B) can start now. If another input transfer follows, it can start after step *g* is complete; however, MSYN cannot be asserted until the conditions of step *c* are met.

2.3.3 DATO and DATOB Bus Transactions (See Figure 2-5)

Because all data transfers are with reference to the master device, a data out (DATO) indicates data transfer from the master to the slave.

a. The master device places the address (or location) of the slave device on the A lines, the data on the D lines, and asserts 2 on the C lines for a DATO or asserts 3 on the C lines for a DATOB.

b. After a 150-ns minimum wait (75 ns to allow for worst case signal skew, and 75 ns to allow internal logic in the skew device to decode the address) the master device asserts MSYN if the bus is inactive (SSYN is clear).

c. The slave decodes A⟨17:00⟩ and, if this is the assigned address of the slave, it responds to MSYN by taking in the data and then asserting SSYN. For a DATO, the slave accepts a full word; for a DATOB, the slave accepts one byte as determined by A00. For consecutive operations with the same slave device, the slave may have to complete an internal action (such as a restore or write cycle) before responding. If the slave is a destructive readout device with the pause flag set (indicating the previous bus transaction was a DATIP), the slave immediately begins a write cycle.

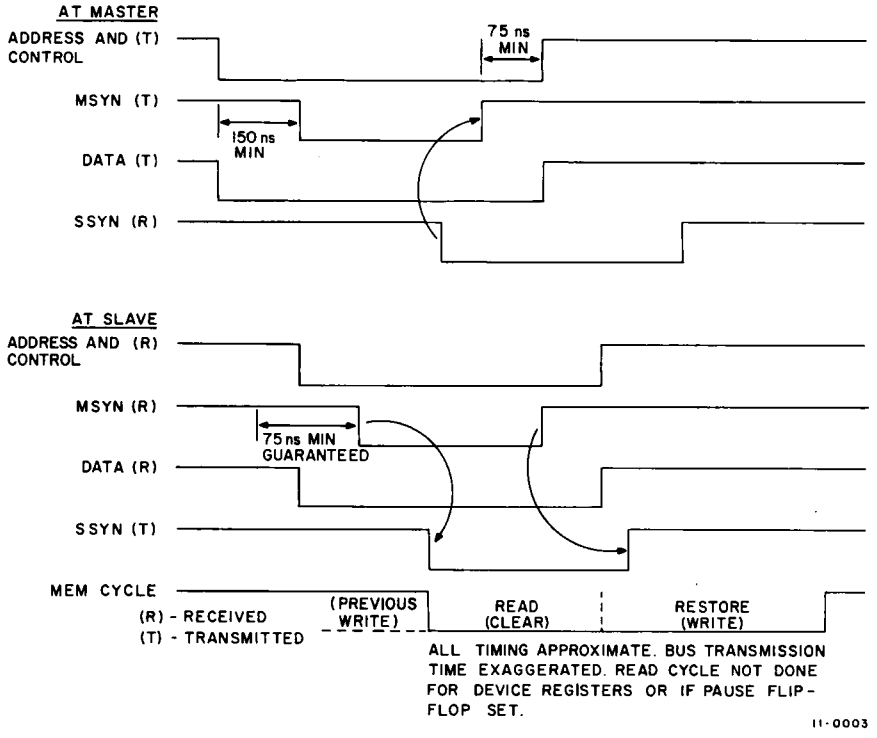


Figure 2-5 DATO or DATOB Timing Diagram

d. The master device receives SSYN and clears MSYN. If the processor is master and SSYN is not received within 20 μs, time out occurs, the MSYN signal is cleared automatically by the processor, and an error trap occurs.

e. After a 75-ns minimum wait, the master clears the A, D, and C lines.

f. The slave device receives cleared MSYN and clears SSYN.

g. The master receives cleared SSYN which signifies the end of the bus transaction. A new bus transaction can begin at step *e* of a DATO or DATOB, and the master can assert MSYN at step *g*, when SSYN is cleared.

2.4 PRIORITY TRANSFER TRANSACTIONS

Transfer of bus control from one device to another is determined by priority arbitration logic, which is part of the processor. Requests for the bus can be made at any time (asynchronously) on the bus request (BR) and nonprocessor request (NPR) lines. During each bus cycle, the arbitration logic first checks for an NPR request (since these requests always take precedence over processor use of the bus). If an NPR is present, the logic issues an NPG signal and receives a selection acknowledge (SACK) signal in return. This procedure occurs simultaneously with the current data transfer. When the device scheduled to become the new bus master is selected, it waits for the present master to clear bus busy (BBSY); then, the newly selected device becomes bus master and asserts BBSY.

A similar procedure occurs at the end of each instruction when the priority arbitration logic checks the bus request lines against the processor priority (as determined by bits ⟨07:05⟩ of the processor status register) and the priority logic issues a grant on the corresponding line. Thus, one of the four levels of BR requests is granted by the processor between instructions unless the instruction currently being executed causes an internal trap (either an error or trap instruction). In this case, BR requests are not granted until completion of the first instruction following the trap sequence. The highest request is always granted first (if the processor priority level is lower than the request level). The grant signals always pass serially through each device connected to the corresponding level in the system. If a device makes a request, it blocks the signal transmission to the next device on the line; otherwise, it passes the signal on.

This causes the device closest to the processor to be the highest subpriority on each request level.

2.4.1 Priority Transfer (PTR) Transaction (See Figure 2-6)

The signal sequence by which a device becomes selected as next bus master is the priority transfer (PTR) bus operation. This operation does not actually transfer bus control; it only selects a device as next bus master. The sequence of events is as follows:

a. The device which needs a data transfer asserts the BR (or NPR) line assigned to it.

b. The processor receives one or more BR signals. These signals enter a priority arbitration system, which compares the NPR and BR levels with the processor priority levels and with each other. If a request has the highest priority entering the arbitration system, and the SACK line is clear, the processor asserts the corresponding BG (or NPG) line. NPG is asserted during the current bus transaction, while BG is asserted only at the end of the current instruction.

c. Each device on the asserted BG line passes the BG signal, unless it is requesting bus control.

d. The first device on the line which has BR asserted responds to the BG by asserting SACK, blocking the BG signal from following devices, and clearing BR.

e. The processor receives the SACK signal and clears BG. (If SACK is not received within 10 μs, time out occurs and the bus grant is cleared automatically by the processor.)

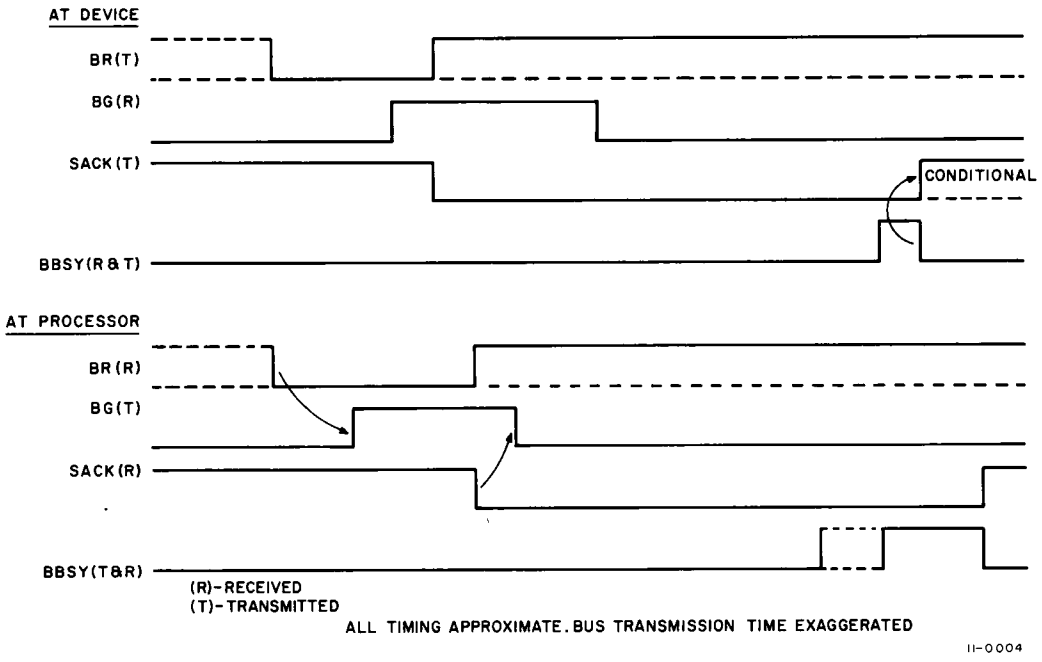


Figure 2-6 PTR Timing Diagram (nominally, for processor master)

f. The current bus master completes a data transfer and clears BBSY at the same time it clears MSYN.

g. The selected device, which is the new bus master, asserts BBSY when BBSY, BG, and SSYN are clear at the end of the previous data transfer. INTR may be asserted at this time, if the new bus master is interrupting. (Refer to Paragraph 2.4.2.)

h. SACK is dropped at the same time INTR is asserted if the device is interrupting. If the device is transferring data, the SACK signal is dropped just prior to the start of the last bus cycle that the device uses.

i. When the new bus master has completed its last data transfer, it clears BBSY. A new bus master then takes control of the bus. If no device is selected (SACK is clear), the processor asserts BBSY and continues processing. If, instead of clearing BBSY in a passive release, the device asserts INTR, the processor conducts an INTR bus transaction. This is called an active release of the bus.

CAUTION

Since an NPR is granted within an instruction, and the interrupt and following processor response would destroy information held in the processor, devices granted bus control through an NPR must not cause a processor interrupt (refer to Paragraph 2.4.2).

NOTE

During the time a master device has control of the bus, it must either issue MSYN or terminate with an interrupt to provide reclocking of the priority arbitration logic in the processor. This affects only certain models of the KA11 and KC11 processors.

2.4.2 Interrupt Transaction (See Figure 2-7)

A device may cause the interrupt operation to occur any time it gains bus control with one of the BR levels. It is usually accomplished immediately on becoming bus master; however, it may follow one or more data transactions on the bus.

a. A device which has been selected as bus master asserts INTR and a vector address on the D lines, at the same time that it clears SACK and asserts BBSY. SACK must remain asserted until INTR is asserted. If the device has been making data transfers prior to the interrupt, it should assert SACK through the last cycle.

b. The processor receives the INTR signal, waits 75 ns for deskew to ensure that all bits of the interrupt vector address are available, and asserts SSYN when the data is read in.

c. The bus master (interrupting device) receives SSYN and clears INTR, the D lines, and BBSY. This constitutes active release of the bus to the processor.

d. The processor clears SSYN when INTR is cleared, and enters the interrupt sequence to store the contents of the current PC and PS registers and replace them with the contents of the location specified by the vector address.

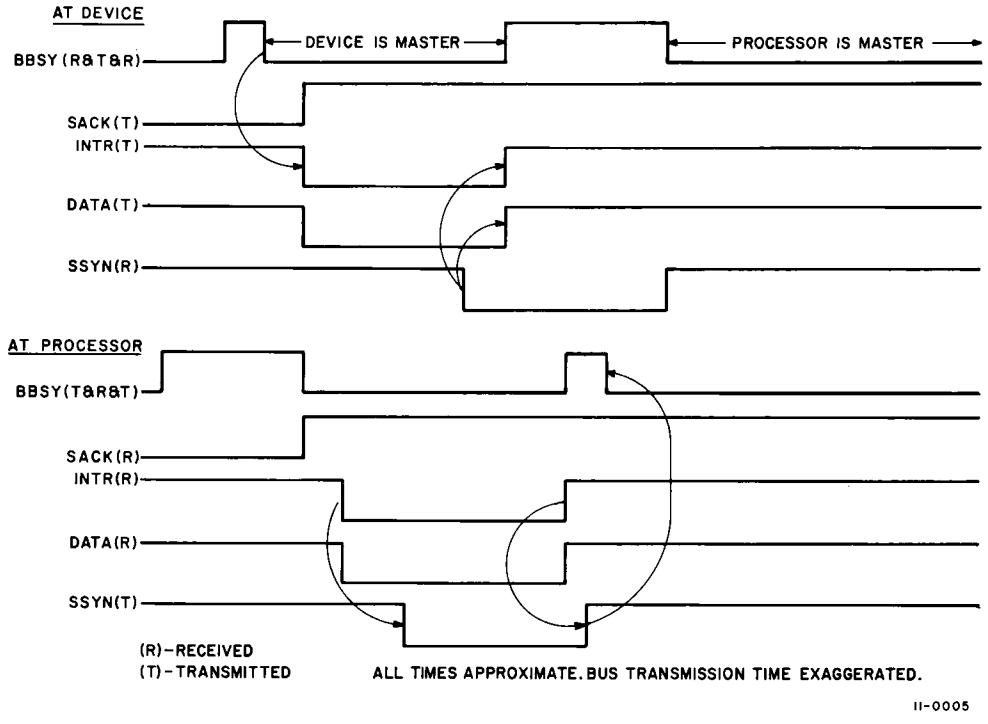


Figure 2-7 INTR Timing Diagram

2.4.3 Priority Chaining

The PDP-11 uses electrical chaining of devices to assign minor priority levels. These levels separate devices of the same major priority level to provide a full array of priority servicing. Figure 2-8 illustrates the mode of operation and advantages of this system. Six devices are shown in order of their electrical distance from the processor. Three devices are at major priority level 4: device A, device C, and device D. The remaining three are at major priority level 5.

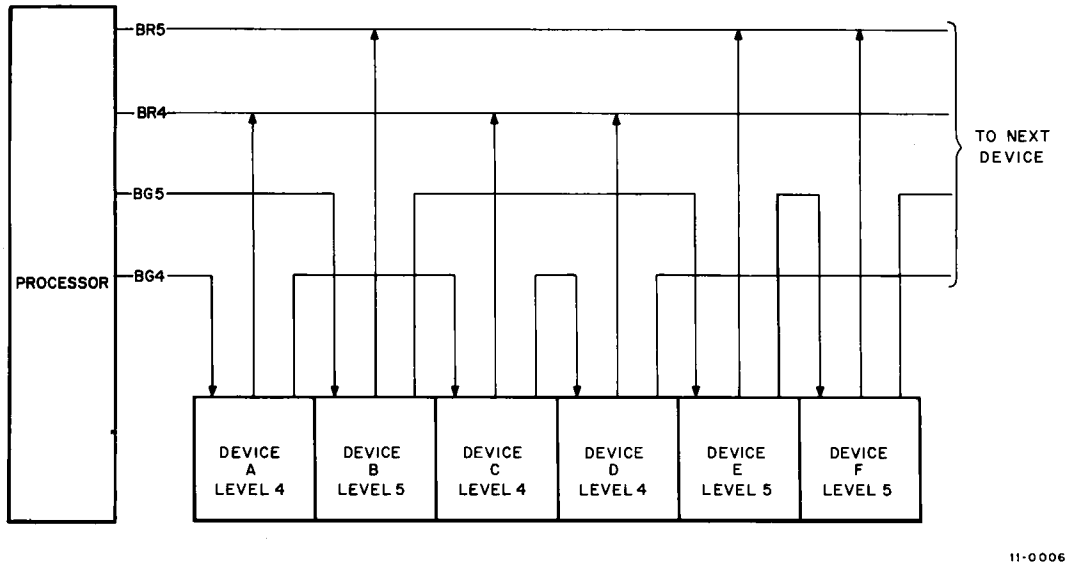


Figure 2-8 Priority Chaining Example

If the processor is at priority level 5 or above, no bus requests are granted from any of these devices. At a processor priority of 4, only requests from devices B, E, or F are granted. Assume that the processor priority is 2 and also that during one instruction cycle, devices C, E, and F assert bus requests. At the end of the instruction, the processor conducts a PTR operation. Since BR 5 is asserted, the processor does not respond to BR 4 (device C). When BG 5 is asserted, the signal first goes to device B. After a delay to clock the interrupt control shift register (discussed in Paragraph 3.3.5), the signal is passed on, since device B was not asserting BR and does not block the pulse. Next, the signal goes to device E, which blocks the pulse, drops BR 5, and takes control of the bus. Device F still has BR 5 asserted, however, and device C has BR 4 asserted. These requests remain on the bus until granted or actively cleared by the processor. If device E does an INTR operation, device F gains control of the bus after the first instruction of the handling routine has been executed, unless the INTR operation raises the processor priority.

Changing the processor priority is accomplished easily since the INTR operation provides a new PS word, which includes a new processor priority. If the priority is set to 5, the processor ignores the current bus requests but grants requests from other devices with higher major priority levels (if there are any).

At the conclusion of the interrupt handling routine, the original processor priority is restored and normal processing is resumed. After one instruction, device F gains control of the bus. When normal processing resumes again, device C, which is still waiting for bus service, gains control in a similar manner.

Higher priorities are assigned to devices that require faster service to avoid destruction or loss of data. Slower devices, which can afford to wait, operate with low priorities. Therefore, service can be provided to all devices in an equitable manner, with no lost data and maximum speed and bus efficiency.

2.5 UNIBUS TIMING

Although all bits of an information signal are transmitted simultaneously, differences in bus path lengths and speeds of individual gate responses may cause variations in transmission time and in the elapsed time before reception. To allow for slow signals to arrive, and to permit settling of levels which have encountered transmission noise, the strobing or gating of this data is delayed a nominal 75 ns. This delay is greater than the worst case signal skew encountered in practice.

A further delay may be necessary to allow an information signal within a device to qualify gates that accept a strobing signal. A 75-ns delay allows for this gating and must be provided by any device which acts as bus master for a data transfer. Thus, a slave is always guaranteed that address and data are valid at its interface (the device side of the receivers) 75 ns in advance of the MSYN signal at the output of the MSYN receiver. If a slave requires more decoding time, it must provide its own delay for the MSYN signal, or trigger a delayed strobe from the MSYN signal.

To simplify slave device design in a DATI or DATIP sequence, the slave may place the data on the D lines coincident with the assertion of SSYN. The deskewing (75 ns) and decoding delay is the master's responsibility. In the INTR sequence, the interrupting device may place the vector address on the D lines coincident with the INTR signal. The processor allows for the 75-ns skew.

2.5.1 Timing Example

To illustrate the operating speed of the Unibus when performing a data transfer, assume a DATO operation to a device that has a flip-flop register. A typical transmitter-bus-receiver delay time is 75 ns. A flow diagram of the transfer procedure is shown in Figure 2-9 and is described below. A simplified logic diagram of the slave device is shown in Figure 2-10.

The bus master places address, control, and data information on the Unibus, waits 150 ns, and then asserts MSYN. After a worst case propagation delay of 75 ns, the slave recognizes MSYN and clocks the data into its register. Since the 75-ns propagation delay *includes* the transmitter and receiver delays, the time required to turn MSYN around into SSYN is literally 0. Also, since the data preceded MSYN by 150 ns (worst case skew is only 75 ns), there is ample preset time for the data input to the slave device flip-flop register.

When the master sees SSYN, it clears its MSYN control (nominal time, 25 ns) and then waits 75 ns before clearing or changing address, control, and data information. Additional cycles may proceed at this time provided SSYN is cleared before MSYN is again asserted. This means that a sustained DATO transfer rate of 400 ns/word may be maintained. This is equivalent to a transfer rate of 40 million bits per second.

Figure 2-11 is a typical timing flow diagram for a DATI transfer. The slave device logic is shown in Figure 2-12. The procedure for a DATI transfer is essentially the same as for a DATO transfer with four exceptions:

a. The master asserts only address and control information.

b. The slave gates data onto the bus simultaneously with the return of SSYN.

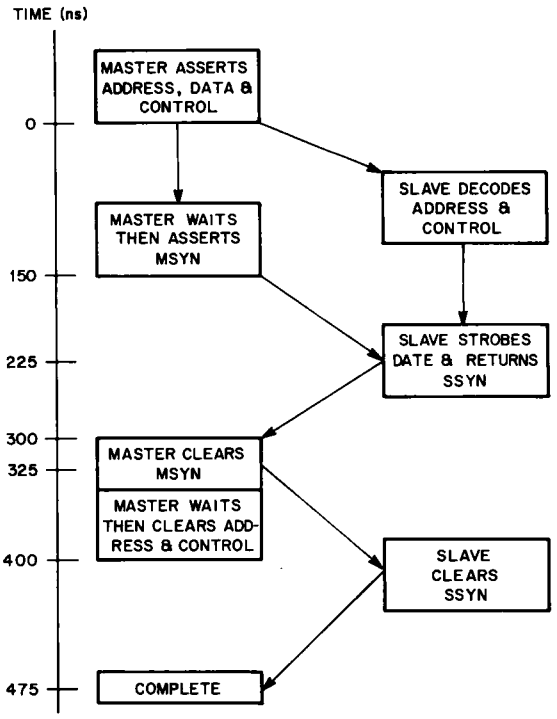
c. The master must wait 75 ns (to allow for skew between SSYN and DATA) before clearing MSYN and strobing data.

d. The slave clears data when it clears SSYN.

The DATI cycle allows sustained transfer rates of 450 ns/word which is equivalent to 35.2 million bits/second.

2.5.2 Time-Out Protection

A precaution must be taken when designing peripheral devices that gain control of the bus for the purpose of transferring data to another element on the Unibus. Normally, such a device contains a bus address register, which is loaded by the program as one of the initialization steps. This address must then be incremented by the device upon completion of each data transfer. If the program loads an erroneous address or if the register increments beyond the available core memory in the existing system, no SSYN response is generated for the data transfer. To prevent this problem from hanging up the system, it is recommended that a 10- to 25-μs integrating one-shot be triggered each time the master device asserts MSYN. If this one-shot times out before SSYN is received, the master should stop the transfer by clearing MSYN, BBSY, and any other signals it has asserted. The master should then set an error flag in its status register.



NOTES: Worst case propagation delay of 75 ns is shown. Total time = 475 ns for a single transfer. Next cycle may begin at 400 ns when master may assert new address and control. Maximum sustained transfer rate is 2.5 million words/sec.

Figure 2-9 Typical DATO Timing Flow

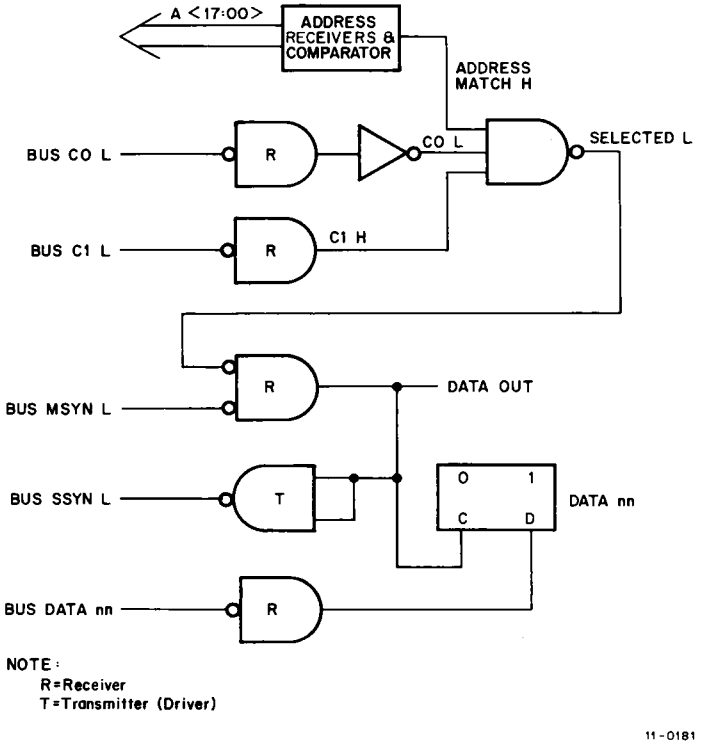
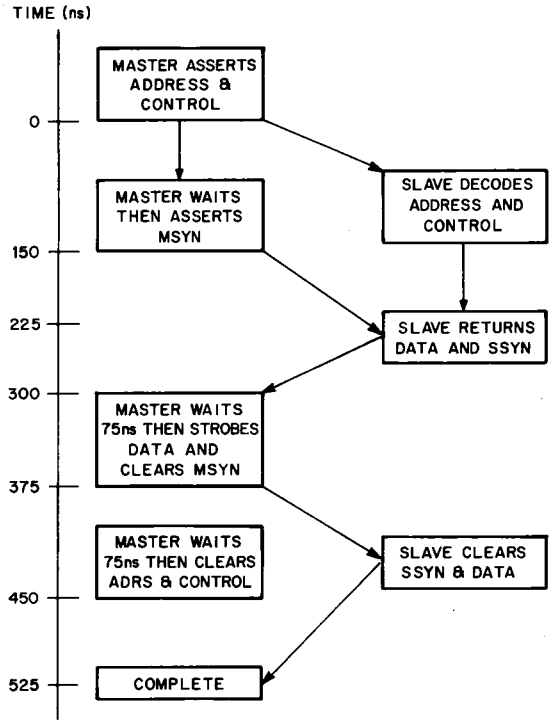


Figure 2-10 Slave Logic For DATO (Fast Response)



NOTES: Worst case propagation delay of 75 ns is shown. Total time = 525 ns for a single transfer. Next cycle may begin at 450 ns when master may assert new address and control. Maximum sustained transfer rate is 2.2 million words/sec.

Figure 2-11 Typical DATI Timing Flow

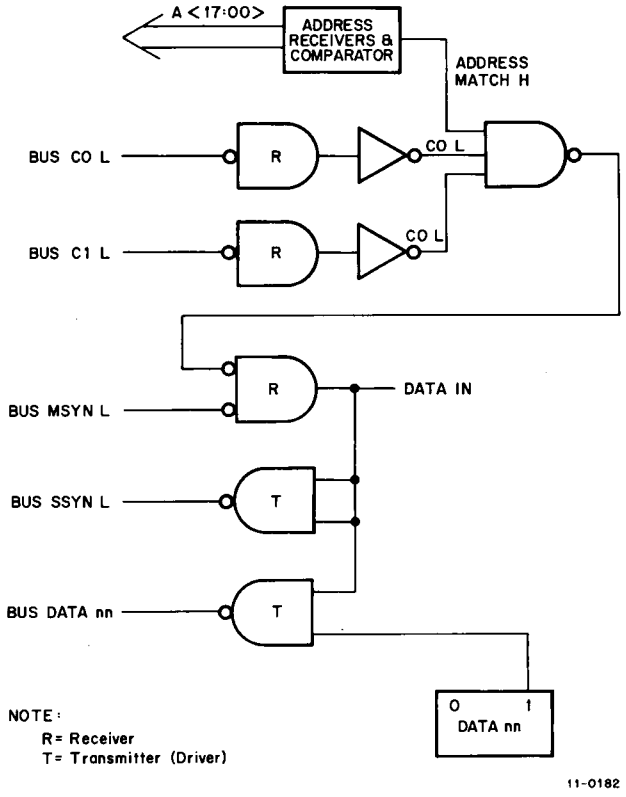


Figure 2-12 Slave Logic for DATI (Fast Response)

2.6 ADDRESS MAPPING

A PDP-11 Address Map is shown in Figure 2-13. Observe that, in the following discussion, all addresses are numbered in octal. The letter K, which is normally used to denote 1000_{10}, is used in this discussion to denote 1024_{10} (2^{10}).

The Unibus addresses 2^{18} locations ($262,144_{10}$ or 256K), and each location contains eight bits. On the basic PDP-11/15 and 11/20 Systems only 16 bits of address information are under program control. This limits the processor to an address map of 64K locations. Since the word length and bus width in the PDP-11/15 and 11/20 are two bytes, most bus operations access two locations at once; the address supplied on the bus is that of the even-numbered location, and the next higher odd location is selected as well. Byte operations can explicitly address any byte. For example, a DATI to location 400 transfers the information in locations 400 and 401, while a DATOB to location 400 loads only location 400. In all cases, a full-word operation cannot address an odd-numbered location.

The address map (Figure 2-13) contains full, 18-bit wide bus addresses. Hardware in the processor forces A(17:16) to ones if A(15:13) are all ones when the processor is master; thus, the last 8K byte locations are relocated to be the highest locations accessible by the bus. All device addresses and internal processor locations are assigned in these 8K locations.

2.6.1 Interrupt and Trap Vector Locations

The first 400 locations in the address map are reserved for trap and interrupt vectors. The stack pointer overflow feature of the processor protects the data in these locations from destruction if the system stack expands downward into this area. Locations 0 through 37 are used for trap vectors for internal processor use, locations 40 through 57 are reserved for use as system software communications words, and the remaining locations (up to location 400) are used for device interrupt vectors. There is no limit to interrupt vectors above 400 except that they are not protected from stack overflow in PDP-11/15 and PDP-11/20 systems.

To prevent customer-designed interfaces from interfering with standard DEC products, the vector addresses (170, 174, 270, and 274) are reserved for customer interfaces.

Each vector requires four locations, and the vector addresses are constrained to even-word boundaries; that is, each vector must end in 4 or 0. (This is implemented by providing vector addresses which do not specify bits 0 or 1. Since the low bits are always 0, address bit 2 specifies either 0 or 4.)

2.6.2 Memory Locations

Memory locations, either read/write or ROM, begin at 0 and proceed to 157777. The highest numbered 8K-block in the map is used by device registers and by internal processor register addresses.

2.6.3 Device Register Locations

Each device has one or more device registers. Device register addresses are always even (A00 is 0), although byte operations may address either half of a register.

The top 8K byte locations are allocated for device register assignment. The top 4K byte (770000–777777) is reserved by DEC for processor addresses and standard peripheral devices. The 2K byte addresses (764000–767777) are reserved for customer allocation and are never assigned by DEC. It is recommended that customer-built interfaces be given addresses in this area.

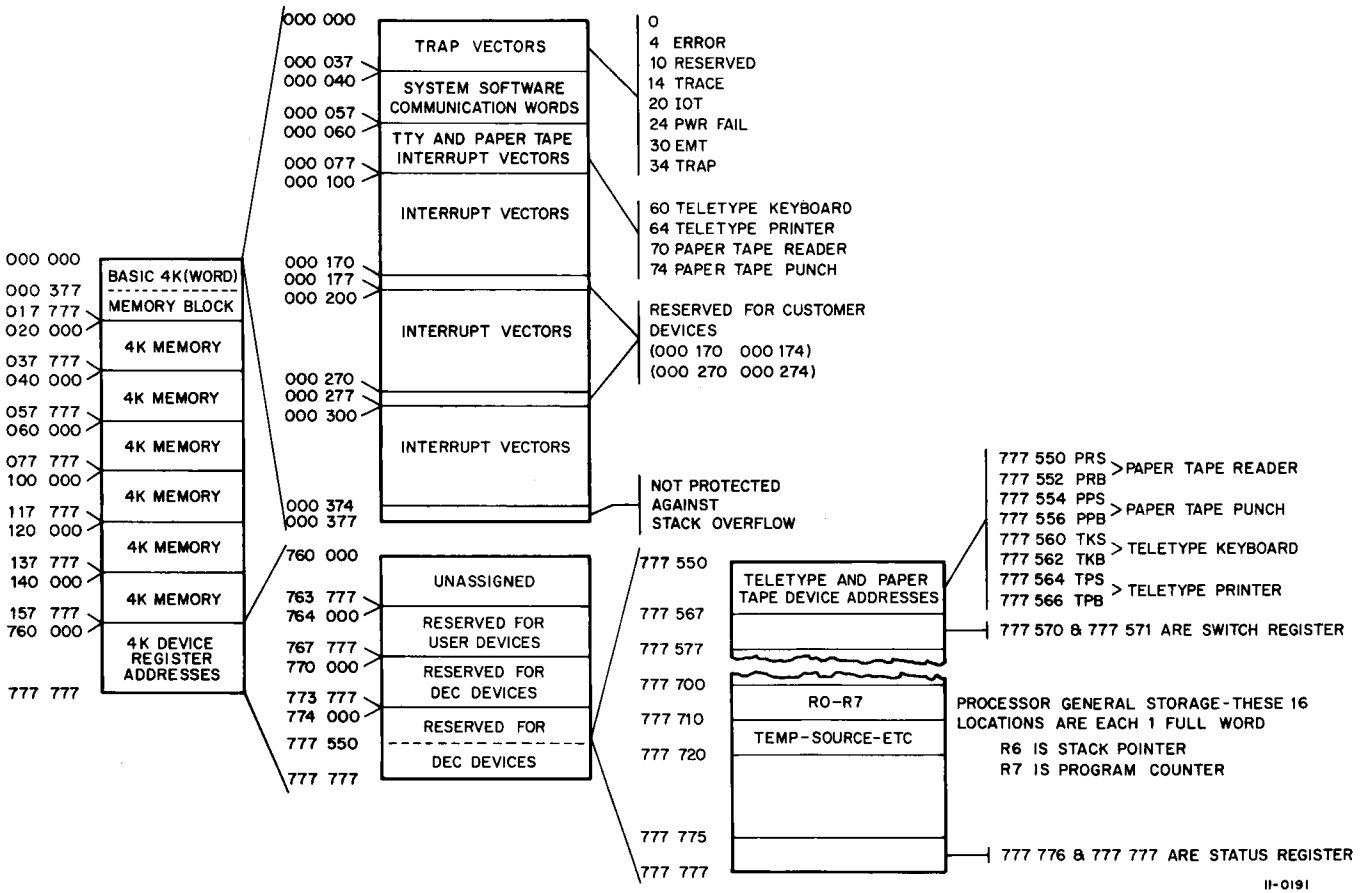


Figure 2-13 PDP-11 Address Map

Starting at location 777550, the first eight locations are reserved for use with the first Teletype and first high-speed paper-tape device. Normally, the PDP-11 System is supplied with the Model 33 ASR Teletype or equivalent device. If a high-speed paper-tape device is used, it is a PC-11 Reader Punch. Specific addresses for each register are shown on the address map (see Figure 2-13).

2.6.4 Processor Locations

Only two processor data locations are explicitly addressable. The console switch register, at locations 777570 and 777571, is a read-only switch register on the PDP-11/20 Console and may be used for program controlled data transfers. The processor status register (PS), at locations 777776 and 777777 contains the processor priority level and the condition codes. The 16 processor storage locations (described in Appendix A) are located at the 16 addresses from 777700 to 777717. Each address accesses a full word of data. These registers are not addressable from the bus, and the addresses are used only for deposit and examine functions from the console.

2.7 DEVICE REGISTERS

The actual transfer of data between a device and the Unibus takes place through one or more registers in the device. These registers may be either flip-flop storage registers or dynamic signals which are simply gated to the bus during a transfer. In addition, it is not necessary for the exact nature of the register bits to be the same. Some bits may be used for read/write (transferred on both DATI and DATO transactions); some may be write only (participate only in DATO transactions, and appear as 0s for DATI's); and some may be read only (participate only in DATI's, unaffected by DATO's). Exercise caution when assigning bit usage. For example, a BIS (Bit Set) instruction to a word containing a write-only bit does not set the bit, but clears it. This is because a BIS performs a DATIP, DATO sequence and, if the bit reads as a 0, it is rewritten as a 0. Examples of all three types are usually found in control and status registers. A typical example of a read/write bit is an interrupt enable bit; an example of the write bit is a go command bit; an example of a read bit is an indicator of an error condition requiring operator intervention.

To standardize register format types, DEC has adopted some preferred bit assignments which are shown in Figure 2-14. The preferred order of register address assignments is given in Table 2-5. These preferences are included for reference only and should not be construed as mandatory requirements for interfacing to the Unibus. The exact nature of register assignments varies with each device. The general philosophy of this is illustrated and discussed in Chapter 4.

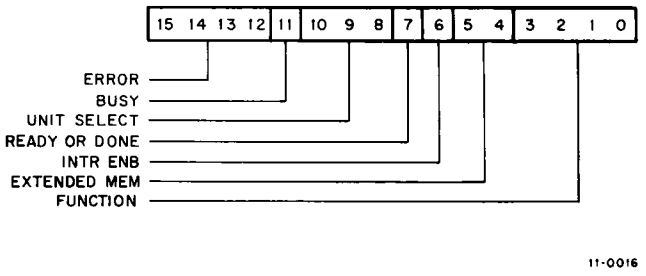


Figure 2-14 Preferred CSR Bit Assignments

FUNCTION: Device Function (read, write, punch, search, etc.) Single function devices should use bit 0 because INC CSR (an operate instruction) performs the command with less program storage and is also faster than a conventional MOV.

EXTENDED MEM: Used to specify A(17:16) when doing device controlled data transfers to locations not in the first 64K block of addresses.

INTR ENB: Interrupt Enable. Inhibits Interrupt on done or error if not set.

READY or DONE: Bit set by device when internal processing is completed and the device is ready to participate in a transfer. Can be checked by the instruction sequence LOOP: TSTB CSR, BPL LOOP.

UNIT SELECT: Used to select multiple devices connected to a single controller (such as DECtape units with operator set unit numbers).

BUSY: Indicates that the device is doing internal processing and cannot participate in a data transfer. Need not be used in many devices, READY may be adequate.

ERROR: Indicates the source or cause of an Error Interrupt. Bit 15 is used for single-error conditions or may be the logical OR of several error conditions to allow the TST instruction to check error status.

NOTE

DONE and ERROR bits are set by the device to cause an interrupt, and must be cleared by the processor to permit future interrupts. Such clearing may be explicit (by transferring 0s to that location) or implicit (by commanding another operation which automatically clears them).

Table 2-5
Preferred Order of Device Register Assignments

ADDRESS (OCTAL)		
N	CSR	CONTROL STATUS REGISTER
N+2	DBR	DATA BUFFER REGISTER
N+4	MAR	MEMORY ADDRESS REGISTER
N+6	WCR	WORD COUNT REGISTER
N+10	DAR	DEVICE ADDRESS REGISTER

CSR – Device function, status and interrupt control.
DBR – Data register for information transfer.
MAR – Memory location for block transfer. Incremented by device logic after each word transfer.
WCR – Set by program to control length of block transfer.
DAR – Track or block number for mass storage devices.

When several registers are used for the same function, they should be assigned contiguous addresses, and be followed by registers of other functions in the same order as for single registers of each function.

CSR1
CSR2
DBR1
DBR2
DBR3
MAR
WCR
DAR

All register types are optional; only implemented registers should be assigned addresses.

CHAPTER 3
INTERFACE CIRCUITS AND HARDWARE

3.1 INTRODUCTION

This chapter discusses the specific circuits, modules, and hardware used for interfacing devices to the Unibus.

3.2 CIRCUITS

The Unibus, a high-speed data transmission facility, imposes certain restrictions when attaching other devices to it. The actual bus is a matched and terminated transmission line which must be received and driven with devices designed for that specific application. The following paragraphs describe bus transmission, bus signal levels, bus length, and bus receiver and transmitter circuits.

3.2.1 Unibus Transmission

The actual bus medium consists of several types of cable. The standard cabling is composed of short jumper modules that interconnect the system units within a mounting box. The M920 Module serves as the jumper module. Critical ground signals are also carried on this module. Cables used between BA11 Mounting Boxes consist of a Flexprint® cable assembly with alternating signals and grounds. The characteristics necessary for proper Unibus transmission are:

Characteristic Impedance	120Ω ±15 percent
Resistance	0.135Ω/ft, maximum

Either twisted pair or coaxial cable laid for minimum crosstalk is recommended for long cable lengths and for applications requiring extreme physical durability of the cable.

The Unibus is terminated at each end by a resistive divider for each signal except the grant signals (see Figure 3-1). The grant signals are terminated with a single resistor. Two M930 Terminator Modules are included in every system to provide these functions.

3.2.2 Unibus Signal Levels

The rest state for all Unibus signal lines, except the grant lines BG⟨7:4⟩ and NPG, is a logic 0 of +3.4V. The asserted state (logic 1) is between ground and +0.8V, which is the saturation voltage of the device driving the bus. The rest state for the grant signals is ground (logic 0) and the asserted state (logic 1) is +3.4V. To guarantee operation under worst case conditions, receivers should have a switching threshold of approximately 2V.

Digital Equipment Corporation uses standard terminology to name signal lines to aid the reader in determining their active state. Either an H or L follows the signal name mnemonic and is separated by a space. This letter

---

®Registered trademark of Sanders Associates.

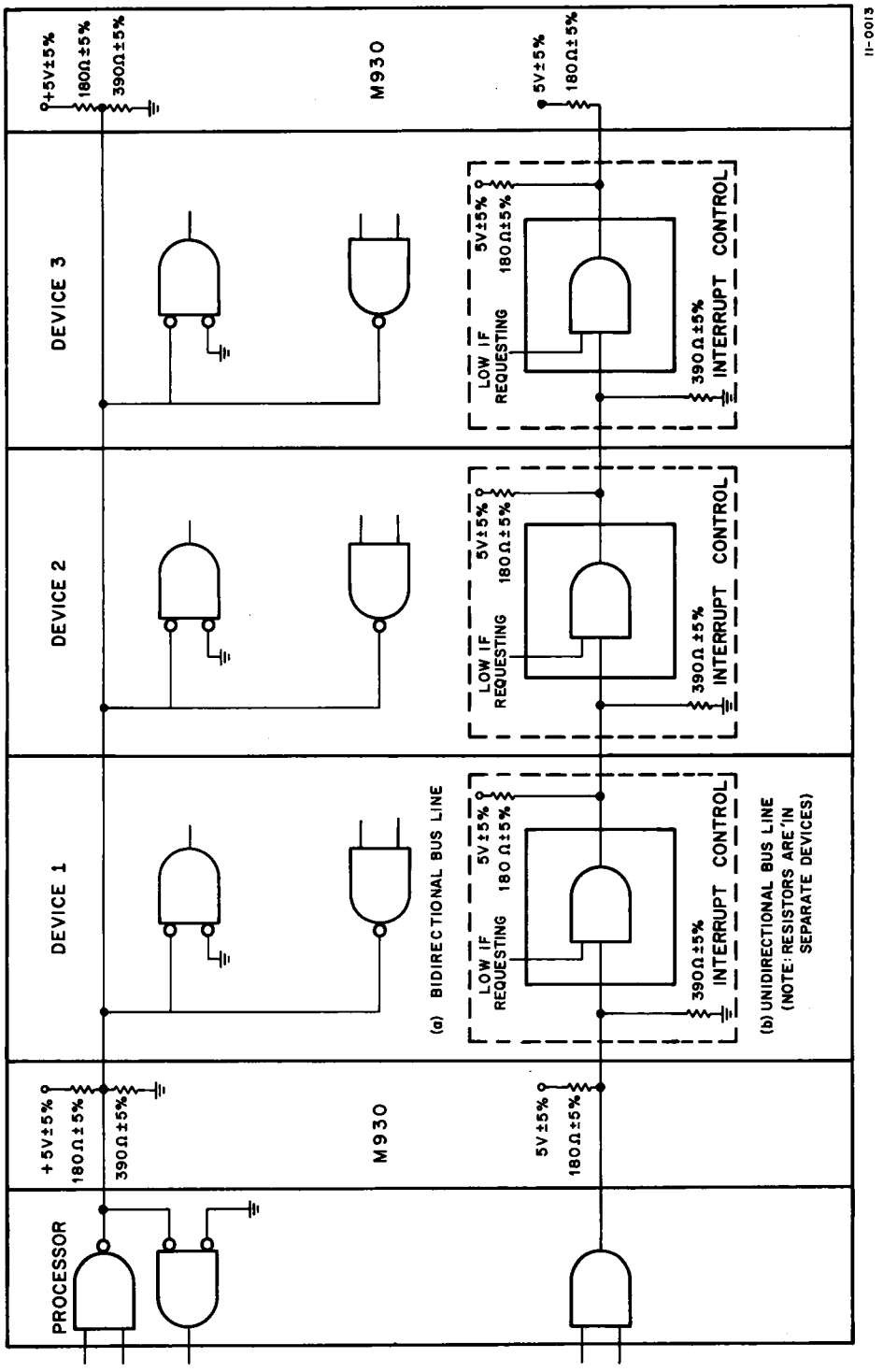


Figure 3-1 Bus Terminations for Bidirectional (a.) and Unidirectional (b.) Bus Lines

indicates the asserted (logic 1) state of the signal to be either high (approximately +3V) or low (ground). Thus, a Unibus data line is called BUS D00 L and a grant line is called BUS BG4 H.

All signals which are not Unibus signals are characterized in terms of standard transistor-transistor logic (TTL) loads. These devices, which are similar to the DEC 7400 Series, have a low state input load of -1.6 mA and a high state leakage current of 40 μA. Outputs are characterized by the number of inputs they can drive (called fanout). A standard TTL gate (as used in the M113) can drive 10 unit loads.

3.2.3 Unibus Length and Loading

The maximum length of the Unibus is a complex relationship involving the type of cable, the bus loading, and distribution of receiver and transmitter taps on the bus. Since the Unibus is a transmission line, and the transfers are asynchronous and interlocked, the electrical delay imposed by length is not a factor.

With Flexprint cable (Tape Cable S-1680), the maximum reasonable length is 50 ft minus the combined length of all stubs or taps, which are those wires from the actual bus to the receivers and transmitters. This maximum length is obtainable only if the individual tap lengths are less than 18 inches and if the loading is not more than a standard of one receiver and two transmitters. If loads are concentrated at one end of the Unibus and a single load is at a distant point, the maximum length could change, provided that the crosstalk of the employed cable is low enough.

The Unibus is limited to a maximum of 20 unit loads. This limit is imposed because of the loading of receivers and leakage of drivers at the high state. This limit is set to maintain a sufficient noise margin. For more than 20 unit loads, a Unibus repeater option (DB11-A) may be used.

3.2.4 Bus Receiver and Transmitter Circuits

The equivalent circuits of the standard Unibus receivers and transmitters are shown in Figure 3-2. Any device which meets these requirements is acceptable. To perform these functions, Digital Equipment Corporation uses two monolithic integrated circuits with the characteristics listed in Table 3-1. Typical transmitter and receiver circuits are shown in Figure 3-3.

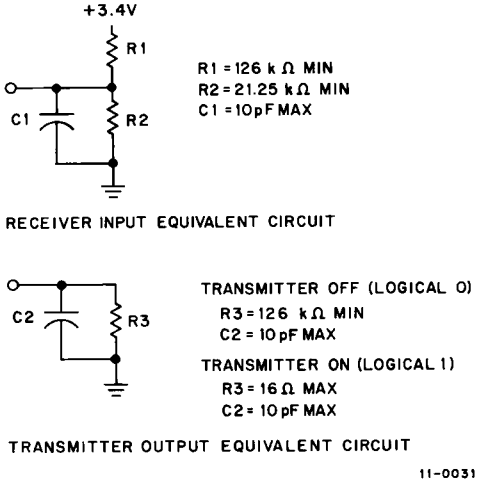


Figure 3-2 Transmitter and Receiver Equivalent Circuits

Table 3-1
Unibus Receiver and Transmitter Characteristics

	Characteristics		Specifications	Notes
Receiver (DEC 380A)	Input high threshold	V_{IH}	2.5V min.	1
	Input low threshold	V_{IL}	1.4V max.	1
	Input current @ 2.5V	I_{IH}	160 uA max.	1, 3
	Input current @ 0V	I_{IL}	±25 uA max.	1, 3
	Output high voltage	V_{OH}	3.5V min.	2
	Output high current	I_{OH}	-2 mA	2, 3
	Output low voltage	V_{OL}	0.6V max.	2
	Output low current	I_{OL}	-12.5 mA	2, 3
	Propagation delay to high state	T_{PDH}	10 ns min. 45 ns max.	4, 5
	Propagation delay to low state	T_{PDL}	10 ns min. 35 ns max.	4, 5
Transmitter (DEC 8881)	Input high voltage	V_{IH}	2.0V min.	6
	Input low voltage	V_{IL}	0.8V max.	6
	Input high current	I_{IH}	60 uA max.	6
	Input low current	I_{IL}	-2.0 mA max.	6
	Output low voltage @50 mA sink	V_{OL}	0.8V max.	1
	Output high leakage current @ 3.5V	I_{OH}	25 μA max.	1, 3
	Propagation delay to low state	T_{PDL}	25 ns max.	5, 7
	Propagation delay to high state	T_{PDH}	35 ns max.	5, 8

NOTES:

1. This is a critical parameter for use on the Unibus. All other parameters are shown for reference only.
2. This is equivalent to being capable of driving seven unit loads of standard 7400 series TTL integrated circuits.
3. Current flow is defined as positive if into the terminal.
4. Conditions of load are 375Ω to +5V and 1.6KΩ in parallel with 15 pf to ground.
5. Times are measured from 1.5V level on input to 1.5V level on output.
6. This is equivalent to 1.25 standard TTL unit loading of input.
7. Conditions of 100Ω to +5V, 15 pf to ground on output.
8. Conditions of 1KΩ to +5V, 15 pf to ground on output.

3.3 UNIBUS INTERFACE MODULES

The following paragraphs describe modules used for Unibus interfacing. These modules include the jumper module, cable, and transmitter and receiver modules that employ the circuits described in the previous paragraph. Descriptions of the address selector, bus and interrupt control, and general device interface modules are also included.

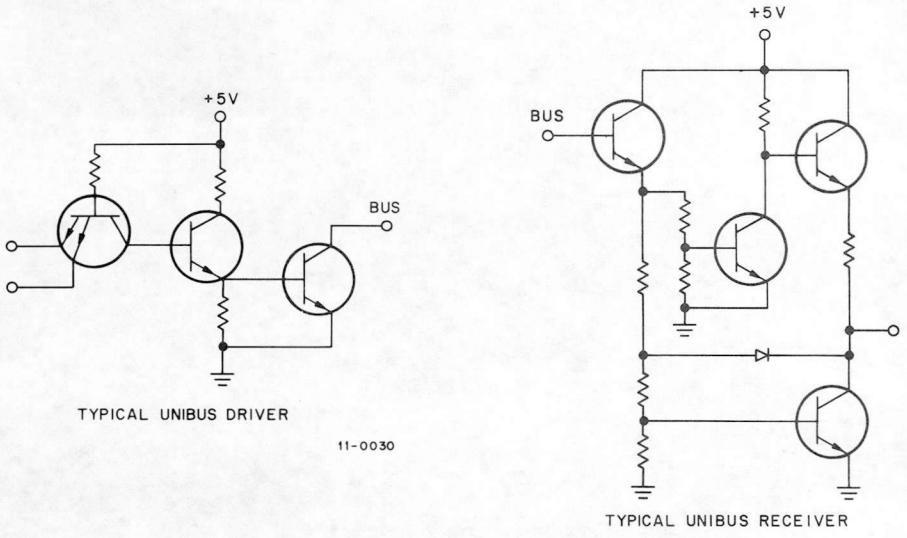


Figure 3-3 Transmitter and Receiver Typical Circuits

3.3.1 Unibus Cables

a. Unibus Jumper Module M920 – The M920 Module (see Figure 3-4) is a double module that connects the Unibus from one system unit to the next. The printed circuit cards are on one-inch centers. A single M920 Module carries all 56 Unibus signals and 14 grounds.

b. Unibus Cable BC11A – The BC11A (see Figure 3-5) is a 120-conductor Flexprint cable used to connect system units in different mounting drawers or to connect a peripheral device removed from the mounting drawer.

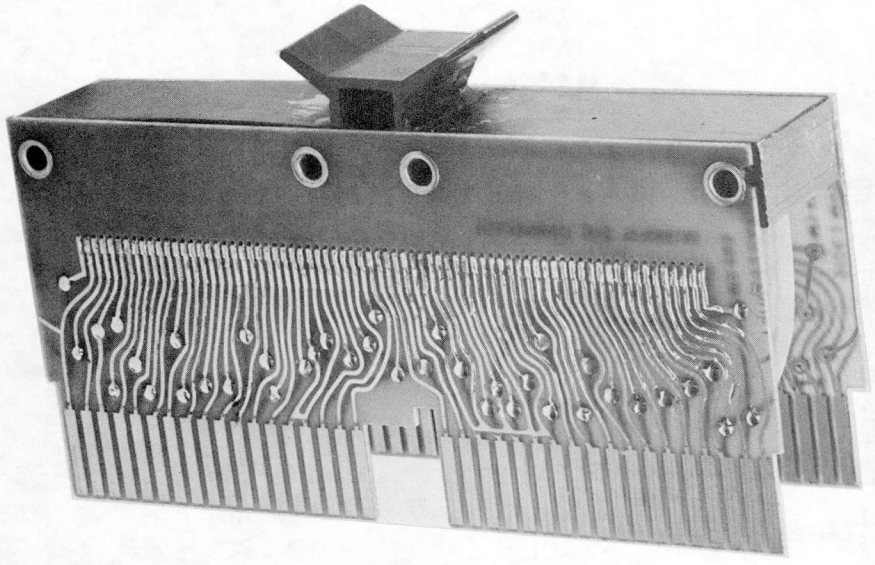


Figure 3-4 Unibus Jumper Module M920

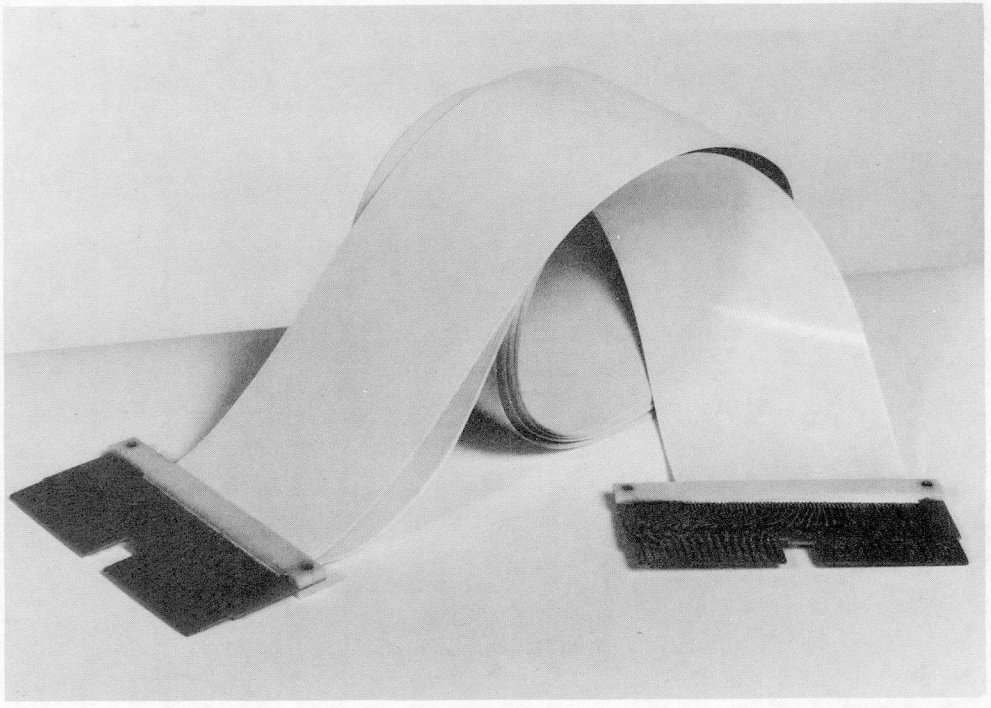


Figure 3-5 Unibus Cable BC11A

The 120 signals include all the 56 Unibus lines plus 64 grounds. Signals and grounds alternate to minimize cross-talk. Cable types and lengths are listed below:

Type	Length (ft.)
BC11A-2	2.0
BC11A-5	5.0
BC11A-8F	8.5
BC11A-10	10.0
BC11A-15	15.0
BC11A-25	25.0

3.3.2 Unibus Terminations

The M930 Unibus Terminator Module is a short, double-size module that terminates all signal lines on the Unibus. This module requires 1.25A at 5V. All pins have a resistive divider termination of 180Ω to +5V and 390Ω to ground, except those listed below:

390Ω in parallel with 0.001 μf to +5 (for AC LO, DC LO)	180Ω to +5V (for grant lines)	Ground Pins		+5V Input Pins
BF1	AV1	AB2	BB2	AA2
BF2	AU1	AC2	BC2	
	BA1	AN1	BD1	BA2
	BB1	AP1	BE1	
	BE2	AR1	BT1	
		AS1	BV2	
		AT1		
		AV2		

3.3.3 Unibus Receivers and Transmitters

Various bus driver and receiver modules may be used in interfaces to provide signal levels compatible with the Unibus. These modules (M783, M784, M785, and M798) are discussed separately in the following paragraphs.

3.3.3.1 M783 Unibus Transmitter – This transmitter module contains 12 drivers; 8 drivers have a common gate line, 4 have 2-input positive AND gating. Input loading is 1.25 standard TTL load. The module is a single-height, 8.5 inch-long Flip-Chip. A circuit schematic of the M783 Transmitter is shown in Figure 3-6.

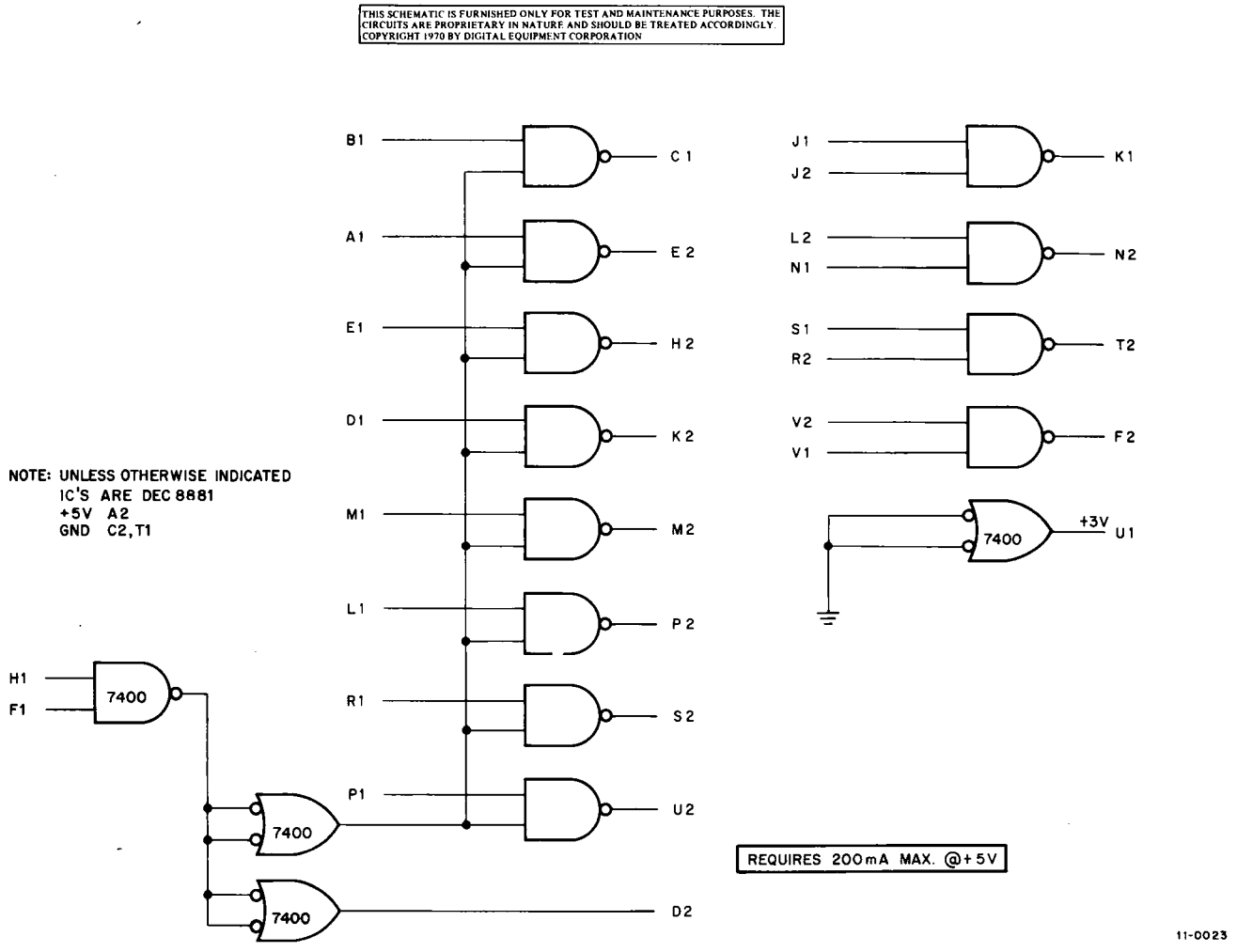


Figure 3-6 M783 Unibus Transmitter (schematic diagram)

3.3.3.2 **M784 Unibus Receiver** – This receiver module consists of 16 DEC 380A inverting circuits which receive bus signals and provide a buffered bus signal output. The output fanout is seven standard TTL unit loads. The receiver module is a single-height, 8.5-inch-long Flip-Chip. A circuit schematic of the M784 Receiver Module is shown in Figure 3-7.

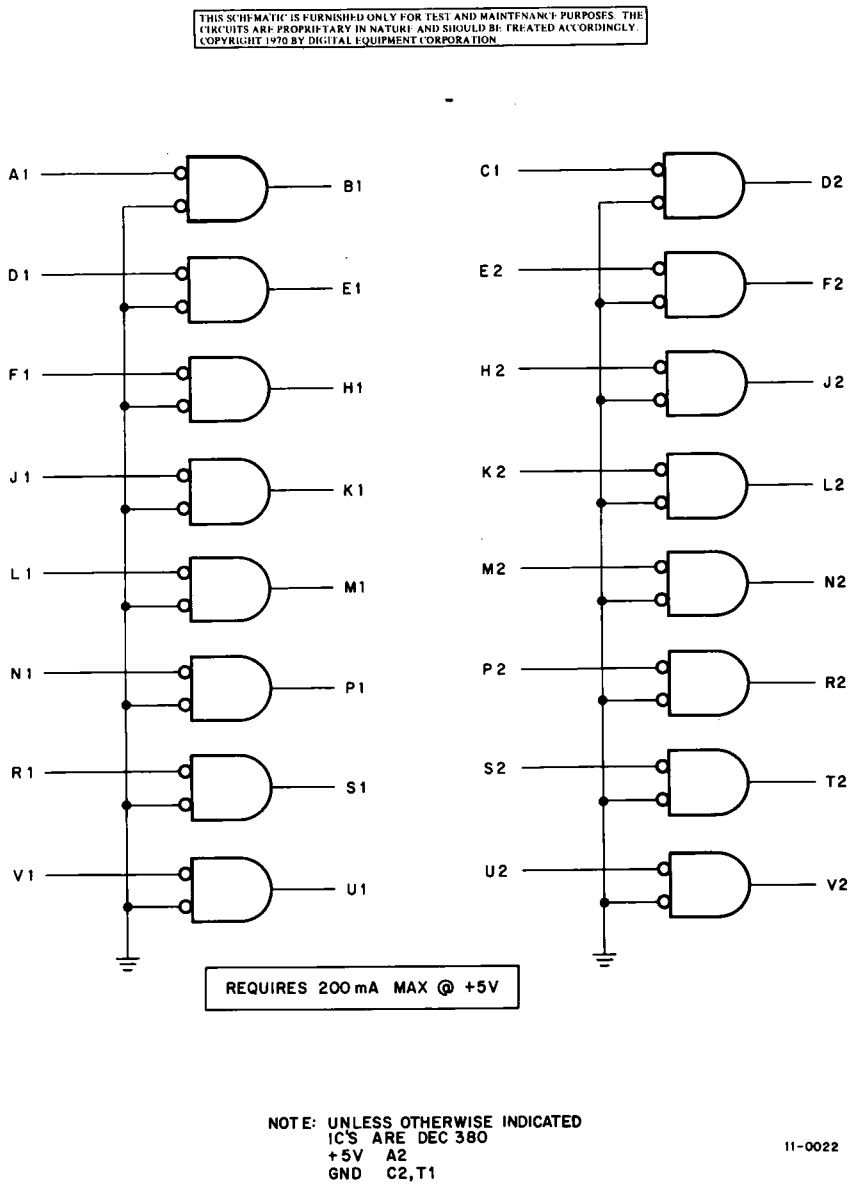


Figure 3-7 M784 Unibus Receiver (schematic diagram)

3.3.3.3 M785 Unibus Transceiver – This module consists of eight pairs of DEC 8881 Drivers and DEC 380 Receivers which are used for bidirectional interfacing to the Unibus. The drivers and receivers have two common gate lines: one for receivers, one for drivers. The driver input loading is 1.25 standard unit load and the receiver fanout is 7 standard TTL unit loads. The module is a single-height, 8.5-inch-long Flip-Chip. A circuit schematic of the M785 Transceiver Module is shown in Figure 3-8.

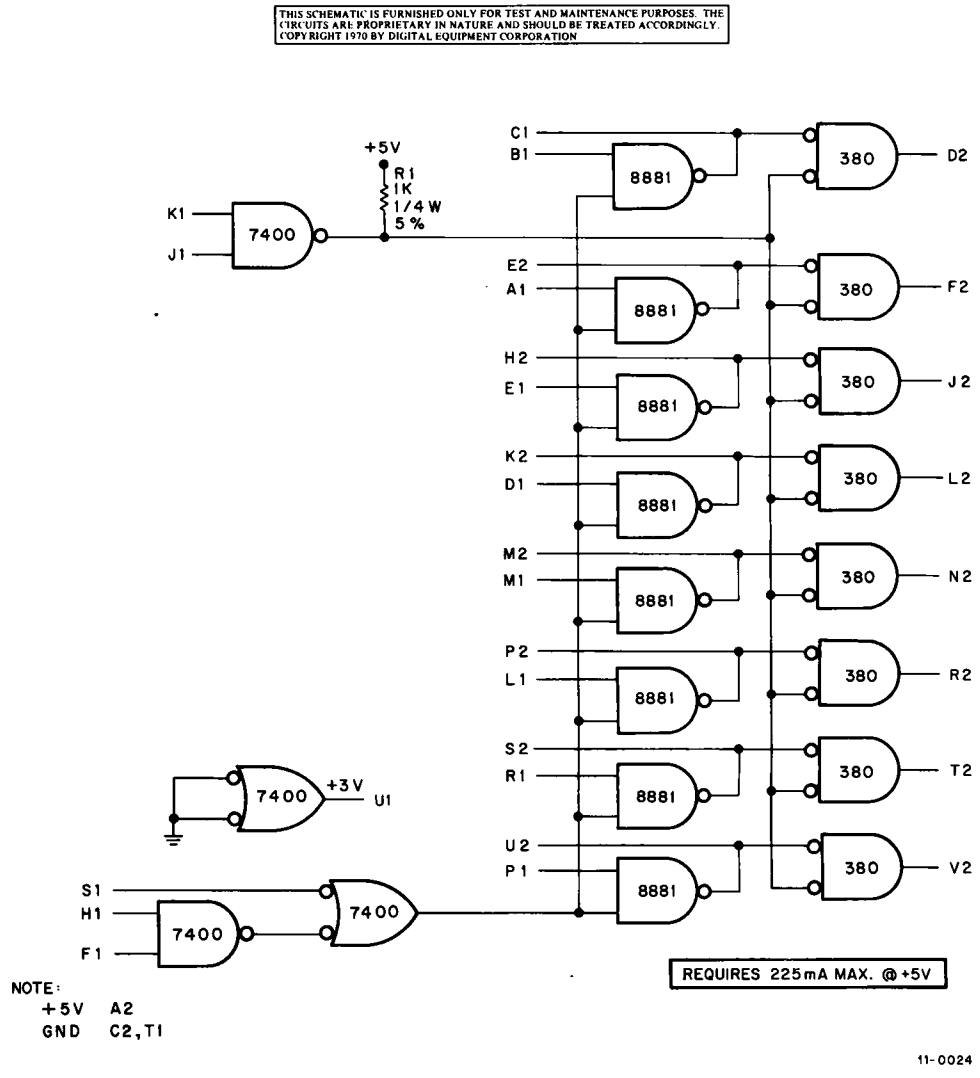


Figure 3-8 M785 Unibus Transceiver (schematic diagram)

3.3.3.4 M798 Unibus Drivers – This module consists of 16 non-inverting Unibus drivers. The module is used in device interfaces to minimize the loading effect caused by attaching several drivers to the same Unibus signal line, as in the case of a device containing multiple registers. A typical driver circuit is shown in Figure 3-9 a. Loading of signal lines on the Unibus is restricted to one unit load per device. A unit load is the equivalent of one receiver input and two driver outputs (210 μA total).

In addition, the M798 module allows the Unibus to be driven by standard open collector TTL gates. The inputs to each M798 driver circuit are pulled to +5V through a 1 kΩ resistor. As shown in Figure 3-9 b, an internal wire-ORed bus is created that is driven from standard open collector gates (available on M141 and M149 modules) or from Unibus drivers (available on M783 Unibus Transmitter and M785 Unibus Transceiver modules). The driver input loading is 4.1 TTL loads for each of the 16 inputs. The output drive signal for each of the 16 outputs is 50 mA sink at 0.8V maximum. The open collector leakage is 25 μA. A circuit schematic of the M798 Unibus Drivers module is shown in Figure 3-10.

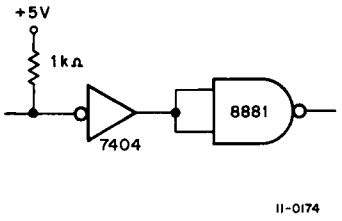


Figure 3-9 a. One of 16 Circuits on the M798

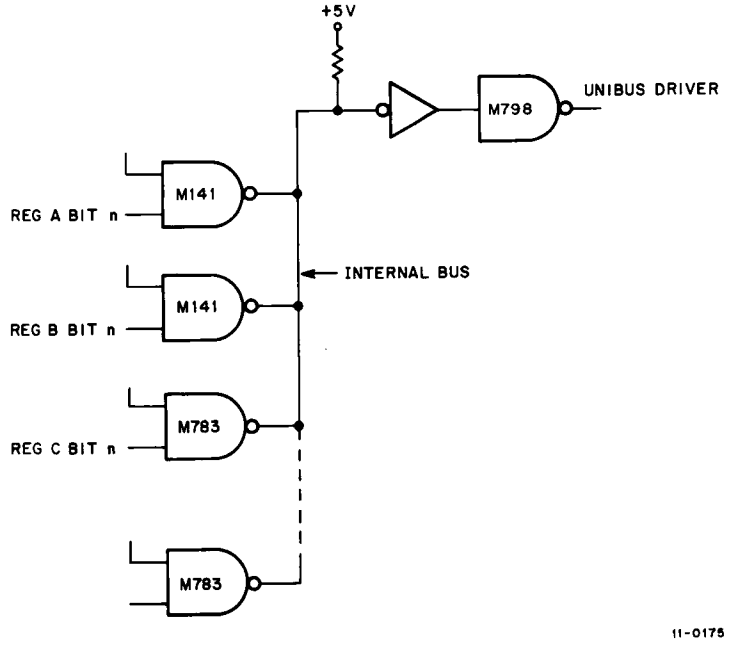


Figure 3-9 b. Typical Use of M798

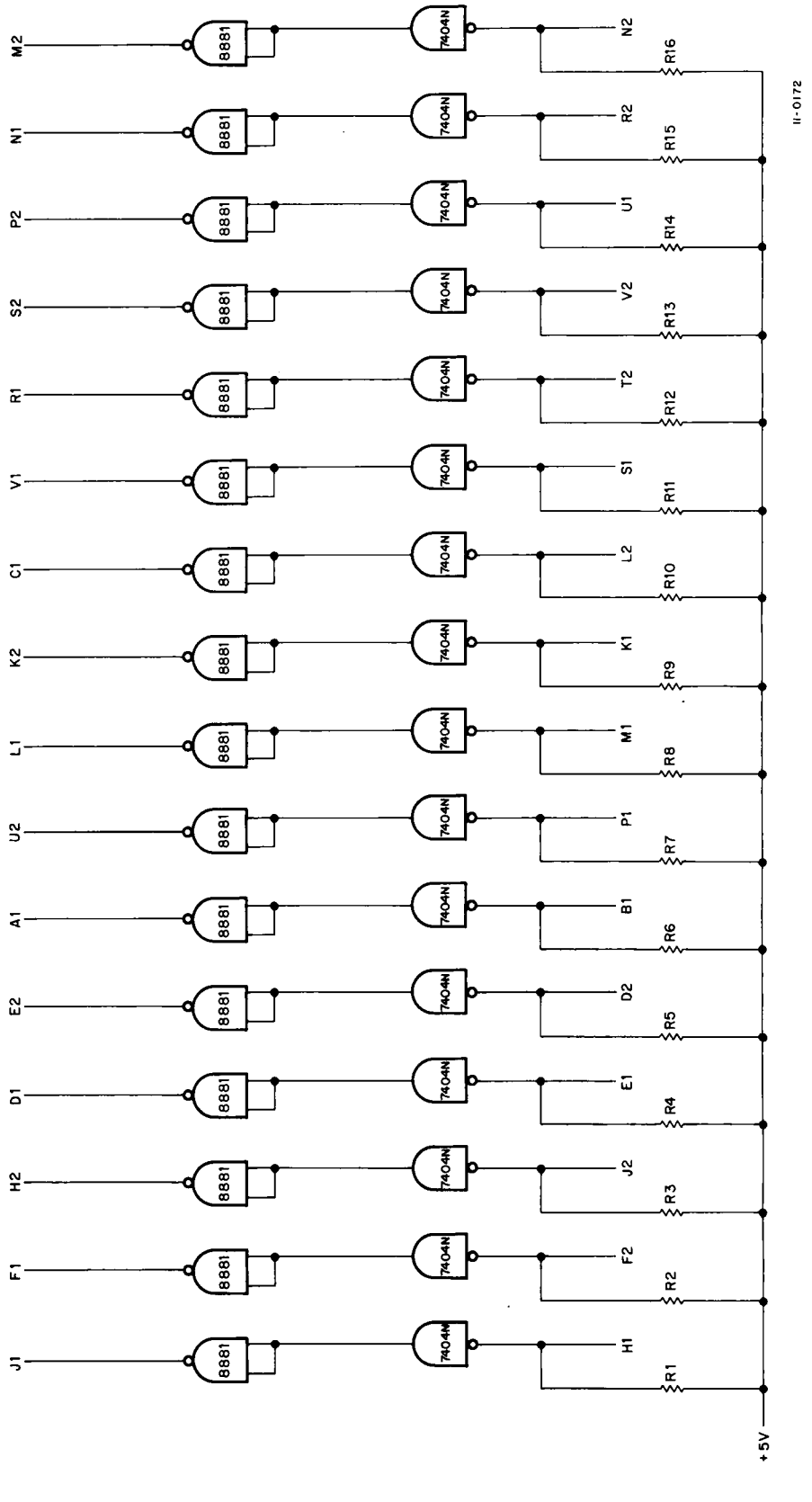


Figure 3-10 M798 Unibus Drivers (schematic diagram)

3.3.4 M105 Address Selector Module

The M105 Address Selector Module provides gating signals for up to 4 full 16-bit device registers. A block diagram of this module is shown in Figure 3-11. Note that IN and OUT are always used with respect to the master (controlling) device. Thus, when the M105 is used in a peripheral device, an OUT transfer is a transfer of data out of the master (such as the processor) and into the device. Likewise, an IN transfer is the operation of the peripheral furnishing data to the processor. The M105 Module is described more fully in following paragraphs.

3.3.4.1 Inputs – The M105 Module input signals consist of 18 address lines, A⟨17:00⟩; 2 bus control lines, ⟨1:0⟩; and a master synchronization ⟨MSYN⟩ line. The address selector decodes the 18-bit address on lines A⟨17:00⟩ as described below. This address format, used for selecting a device register, is shown in Figure 3-12. Note that all inputs are standard bus receivers.

a. Line A00 is used for byte control.

b. Lines A01 and A02 are decoded to select one of the four addressable device registers.

c. Decoding of lines A⟨12:03⟩ is determined by jumpers on the module. When a given line contains a jumper, the address selector searches for a zero on that line. If there is no jumper, the address selector searches for a one.

d. Address lines A⟨17:13⟩ must be all ones. This specifies an address within the top 8K byte address bounds for device registers.

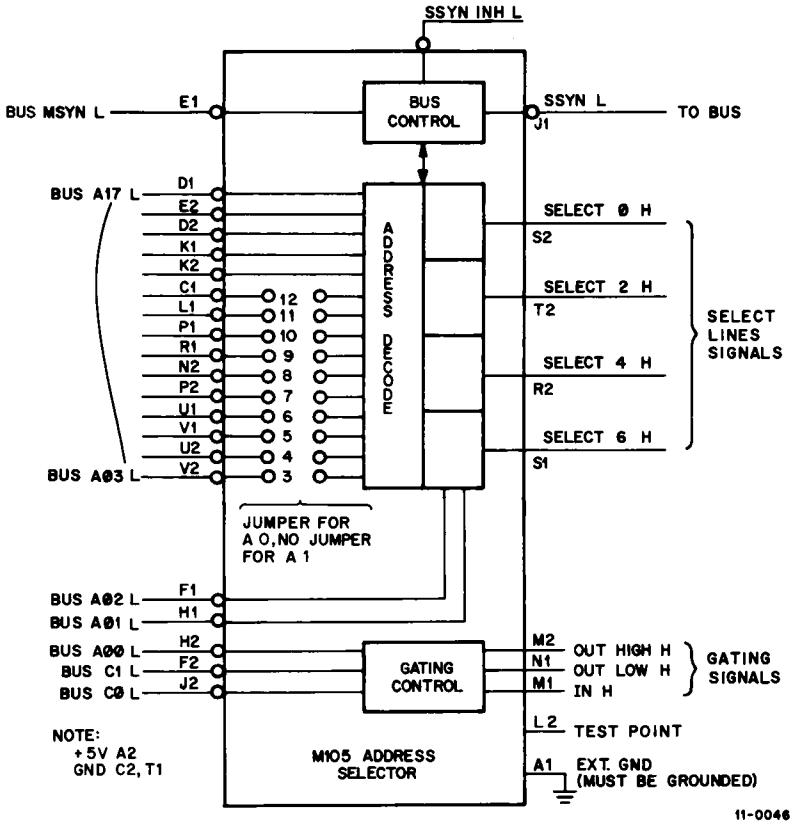


Figure 3-11 M105 Address Selector

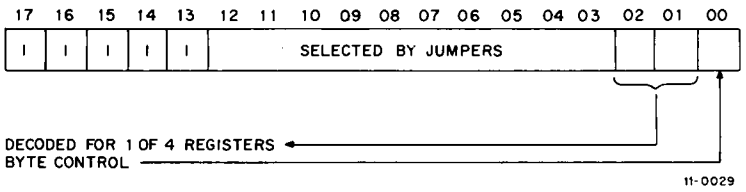


Figure 3-12 Device Register Select Address Format

3.3.4.2 **Slave Sync (SSYN)** – When SSYN INH is grounded, it inhibits the acknowledgment signal (SSYN) normally generated by the M105. In this case, the SSYN must be generated by another source. When SSYN INH is not grounded, SSYN is returned to the master 100 ns after register select becomes true. This time may be extended to a maximum of 400 ns by adding an external capacitor between SSYN INH and ground.

3.3.4.3 **Outputs** – The M105 Output Signals permit selection of four 16-bit registers and provide three signals used for gating information to and out of the master device. The M105 may be used instead to select up to eight 8-bit registers, or any appropriate combination of byte and word registers.

The input signals select the M105 control output line states as shown in Tables 3-2 and 3-3.

Table 3-2
M105 Select Lines

Input Lines A⟨02:01⟩	Select Lines True (+3V)
00	0
01	2
10	4
11	6

NOTE
1. Lines A⟨17:13⟩ must be all 1s (0V on Unibus).
2. Lines A⟨12:03⟩ are selected by jumpers.

Table 3-3
Gating Control Signals

Mode Control C⟨1:0⟩	Byte Control A00	Gating Control Signals True (+3V)	Bus Sequence
00	0	IN	DATI
00	1	IN	DATI
01	0	IN	DATIP
01	1	IN	DATIP
10	0	OUT LOW OUT HIGH	DATO

THIS SCHEMATIC IS FURNISHED ONLY FOR TEST AND MAINTENANCE PURPOSES. THE CIRCUITS ARE PROPRIETARY IN NATURE AND SHOULD BE TREATED ACCORDINGLY.
COPYRIGHT 1970 BY DIGITAL EQUIPMENT CORPORATION

REQUIRES 340 mA MAX. @ +5V

NOTE:
UNLESS OTHERWISE INDICATED
RESISTORS ARE 1K, 1/4W, 5%
+5V A2
GND C2, T1

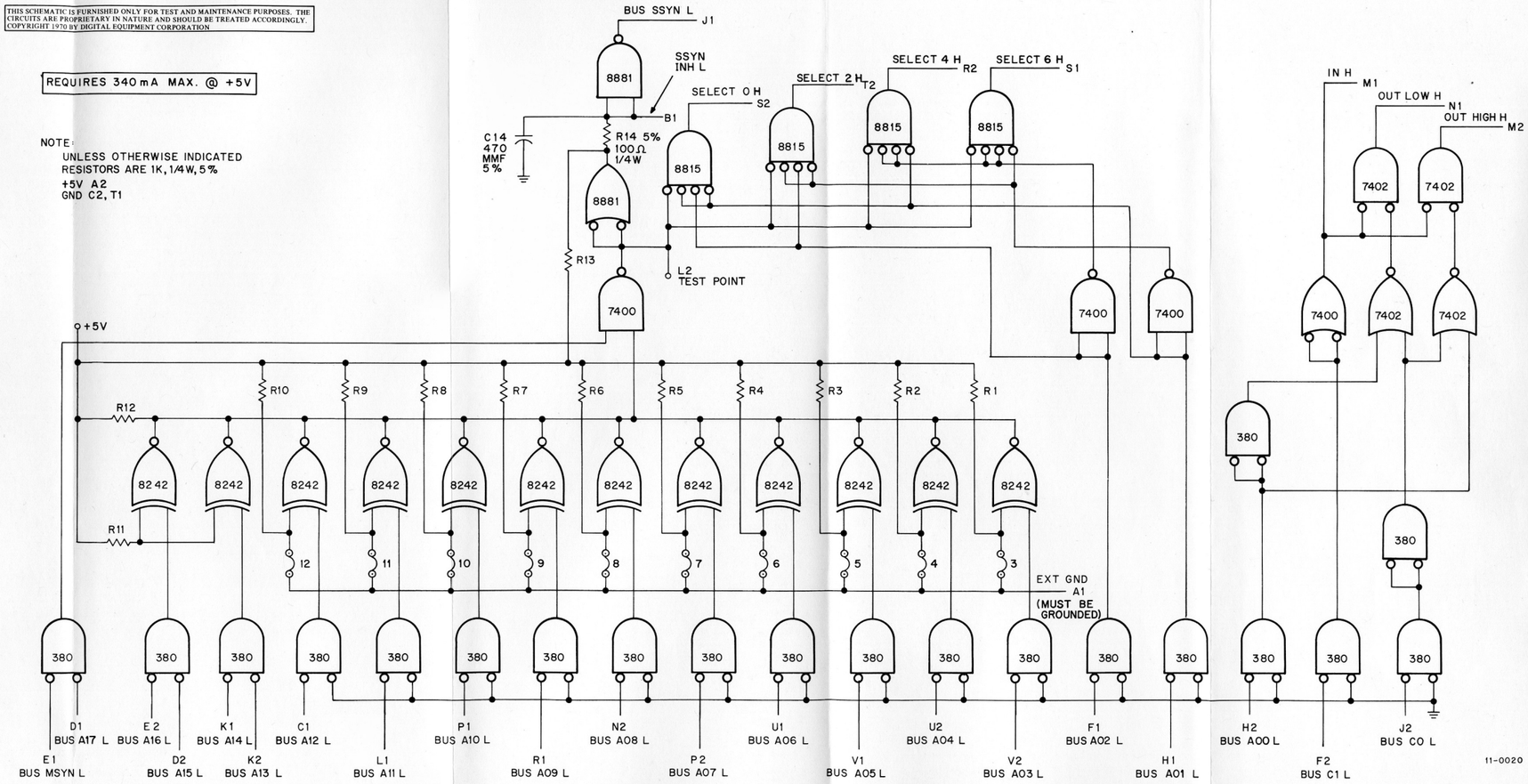


Figure 3-13 M105 Address Selector (schematic diagram)

Table 3-3 (Cont)
Gating Control Signals

Mode Control C⟨1:0⟩	Byte Control A00	Gating Control Signals True (+3V)	Bus Sequence
10	1	OUT LOW OUT HIGH	DATO
11	0	OUT LOW	DATOB
11	1	OUT HIGH	DATOB
NOTE Gating control signals may become true although select lines are not.			

3.3.4.4 **Specifications** – The M105 output fanout is ten standard TTL loads for register select lines and eight standard TTL loads for gating control lines. The module is a single-height, 8.5-inch-long Flip-Chip. A circuit schematic for this module is shown in Figure 3-13. Note that pin A1 (EXT GND) must be grounded by the user.

3.3.5 M782 Interrupt Control Module

The M782 Interrupt Control Module provides the circuits and logic required to make bus requests and to gain control of the bus (become bus master). The module also includes circuits needed to generate an interrupt, if desired. The module contains two completely independent request and grant acknowledge circuits (channels A and B) for establishing bus control. The interrupt control circuit can be used with either, or both, of the request channels and provides a unique vector address for each channel. Figure 3-14 is a block diagram of the M782 Module, which is a single-height, 8.5-inch-wide Flip-Chip.

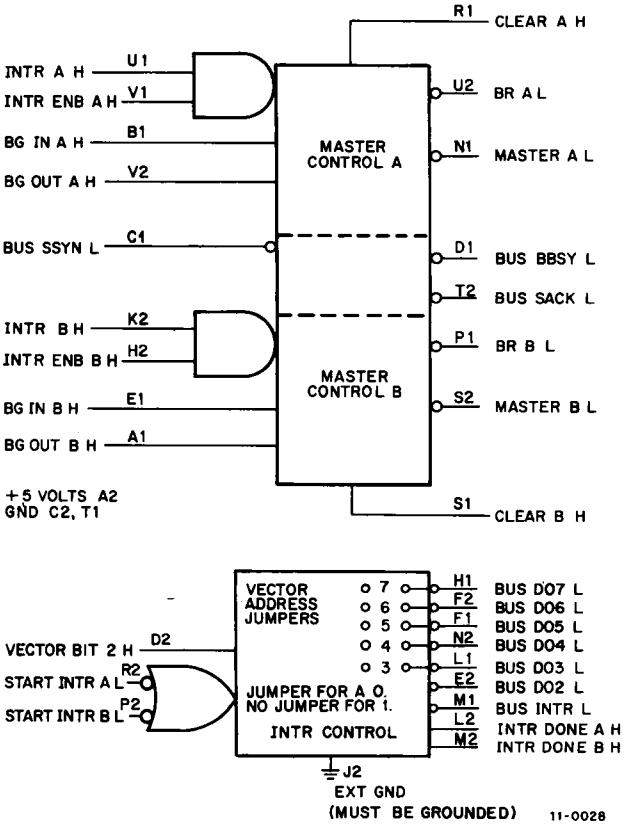


Figure 3-14 M782 Interrupt Control (block diagram)

The master control section (either channel A or B) is used to gain control of the bus. When the INTR and INTR ENB requesting inputs are asserted, a bus request is made on the BR level corresponding to the level of the BR line wired to the BR pin of the module. When the priority arbitration logic in the system recognizes the request and issues a bus grant signal, the master control circuit acknowledges with a SACK signal. When the device has fulfilled all requirements to become bus master, the master control circuit asserts BBSY and then asserts a MASTER signal. (Refer to Section 2.4.1.)

Once the device has gained bus control by means of a BR request, an interrupt can be generated. If an interrupt is desired, the module is interconnected as shown in Figure 3-15. This figure illustrates the use of the two channels to first generate requests for bus control and then initiate interrupts. The request from channel A is a slightly higher priority than the channel B request because the bus grant signal first enters A, then enters B.

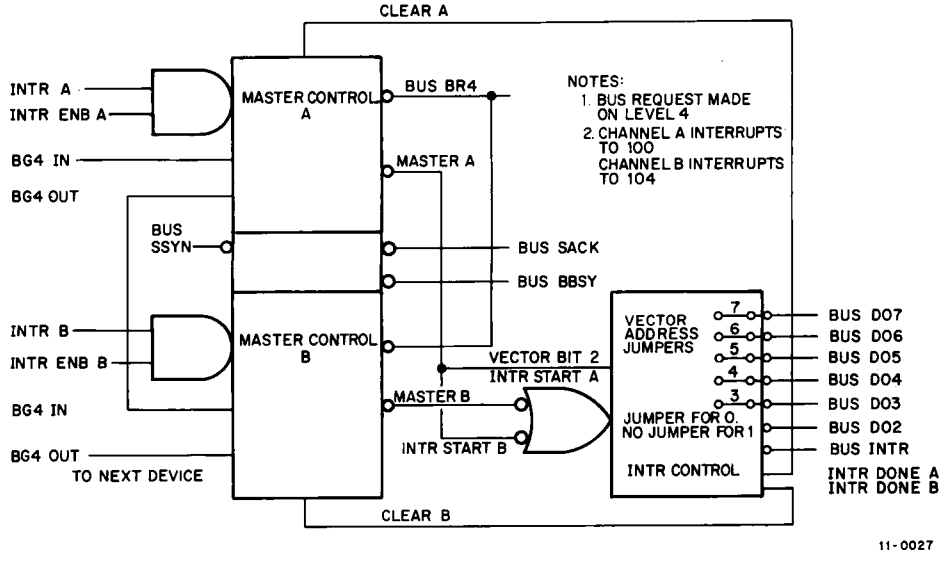


Figure 3-15 M782 Interconnection for 2-Channel Interrupt

The vector address is selected by jumpers on the M782 Module. Since the vector is a two-word (four-byte) block, it is not necessary to determine the state of bits 0 and 1. The six selectable lines determine the vector address. The least significant line is controlled by the VECTOR BIT 2 input signal. If this input is asserted, then bus line D02 is asserted. Thus, the interrupt on channel A uses a vector at location 100 and channel B uses a vector at location 104.

Figure 3-16 illustrates an M782 Module used for bus control in a device that directly transfers data to memory and then causes an interrupt when the transfer is completed. Channel A is connected to the NPR and NPG lines and is used to gain bus control for direct to memory, or device-to-device, transfers. Channel B is used to gain bus control for an interrupt.

Each M782 Module master control section contains two flip-flops that sequence through four states, thereby controlling the request for bus control. Figure 3-17 is a state diagram of this sequence and Figure 3-18 shows a circuit schematic of the M782. The BG IN signal is allowed to pass through the module to BG OUT when the device is not issuing a request (state A), is master (state D), or has had the request honored (state E). To request bus use, the AND condition of INTR and INTR ENB must be satisfied. These levels must be true at least until the request is granted. Once bus control has been attained, it can be released by either asserting CLEAR or by negating either INTR or INTR ENB. The first method leaves the master control in state E, thereby inhibiting further bus requests

even if INTR and INTR ENB remain asserted. In order to make another bus request, INTR or INTR ENB must be dropped and then reasserted to cause the module to advance from state E through state A to state B where it asserts the request line. This prevents multiple interrupts when the master control is used to generate interrupts. The second method is used to release the bus after NPR use. Note that pin J2 (EXT GND) must be grounded by the user. A summary of all M782 signals is listed in Table 3-4.

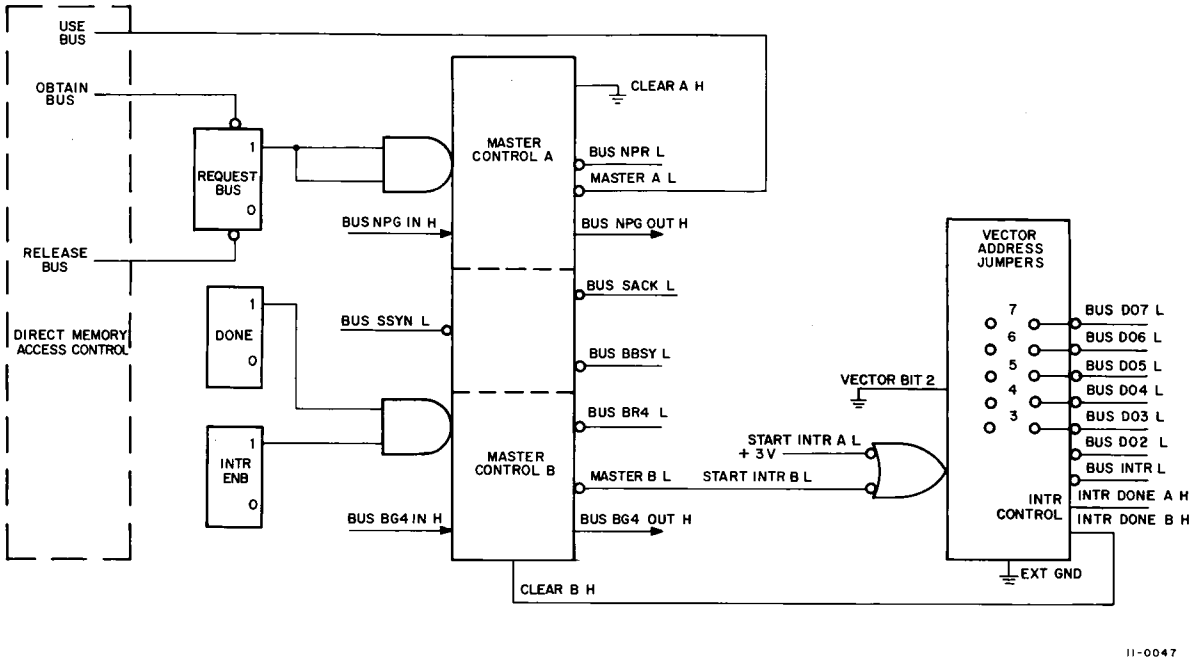


Figure 3-16 M782 Interconnection for Direct Memory Access

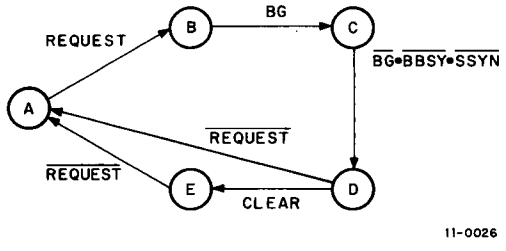


Figure 3-17 State Diagram of Master Control

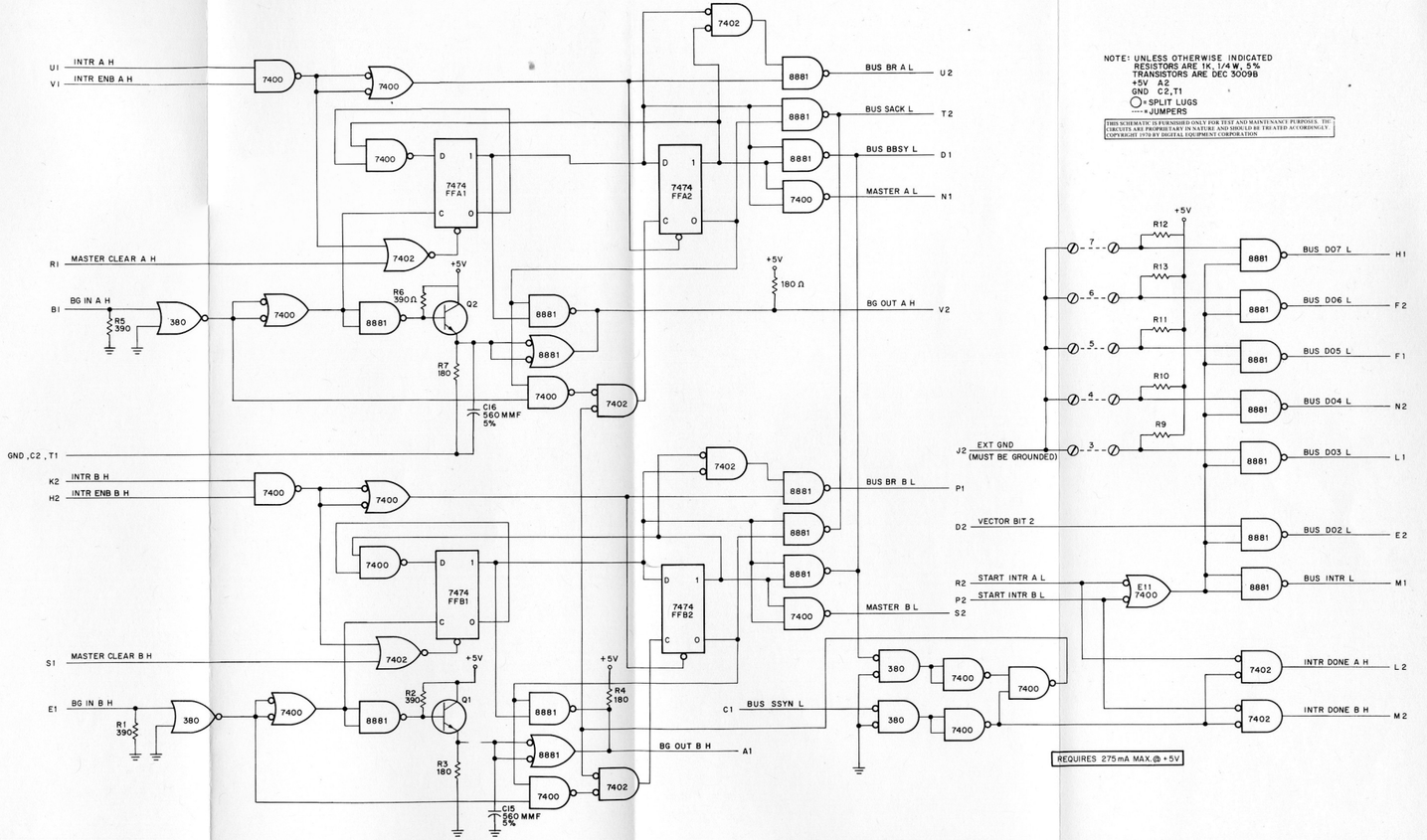


Figure 3-18 M782 Interrupt Control (schematic diagram)

Table 3-4
Summary of M782 Signals

Signal	Assertion Level	Input Loading	Output Drive
INTR A, B	H	1 TTL (each)	
INTR ENB A, B	H	1 TTL	
CLEAR A, B	H	1 TTL	
MASTER A, B	L		10 TTL
START INTR A, B	L	2 TTL	
INTR DONE A, B	H		10 TTL
BG IN A, B	H	1 R*	
BG OUT A, B	H		2 D**
BR A, B	L		1 D
VECTOR BIT 2	H	1 TTL	
BUS SSYN	L	1 R	
BUS BBSY	L	1 R	2 D
BUS SACK	L		2 D
BUS INTR	L		1 D
BUS D⟨07:02⟩	L		1 D

*R = Standard Unibus receiver load.
**D = Standard Unibus transmitter (driver) output.

3.3.6 M795 Word Count and Bus Address Module

The M795 Word Count and Bus address Module is used to interface direct memory access (DMA) devices to the PDP-11 Unibus. This module contains two 16-bit counters: one counter is used to count the number of data transfers that occur; the other counter is used to specify the bus address of the data to be transferred.

Block transfer devices that function as bus master during data transfers usually require two registers to hold the parameters of the transfer. One parameter is transfer count. Initially, a register is loaded with the 2's complement of the number of items to be transferred to or from memory. After each transfer is complete, the register is incremented. If the new value of the register is 0 (indicated by an overflow), further transfers are inhibited and the block transfer is complete. Since information can be transferred in words (16 bits each) on the Unibus, the name Word Count (WC) is usually assigned to this register. However, the Unibus is also capable of transferring 8-bit bytes of data at a time, and this register may be used equally as well as a Byte Count register.

The second parameter used in block transfers is the transfer address. Initially, a register is loaded with an address that specifies the memory location to, or from, which data is to be transferred. The register is incremented after each transfer; thus, the register continually "points" to sequential memory locations. Since memories and devices have addresses on the Unibus, this register is usually called the Bus address (BA) register.

A simplified block diagram of the M795 module is shown in Figure 3-19. Both the word count (WC) and bus address (BA) registers consist of 16 flip-flops. These flip-flop registers can be loaded by placing data on the 16 data line inputs common to both registers and asserting the appropriate loading signal. There are four independent loading signals: WC high byte, WC low byte, BA high byte, and BA low byte. Each of the outputs of the 16 bits in the WC register are connected to a set of DEC 8881 Unibus drivers. The contents of the WC register can be gated to the data bus when the appropriate gate signal is activated. The BA register also has a set of Unibus drivers connected to each output so that the register contents can be gated to the data bus. Note that the driver outputs of both the WC and BA registers are wire ORd together. In addition, the BA register has a set of drivers with independent outputs to allow it to drive the address bus.

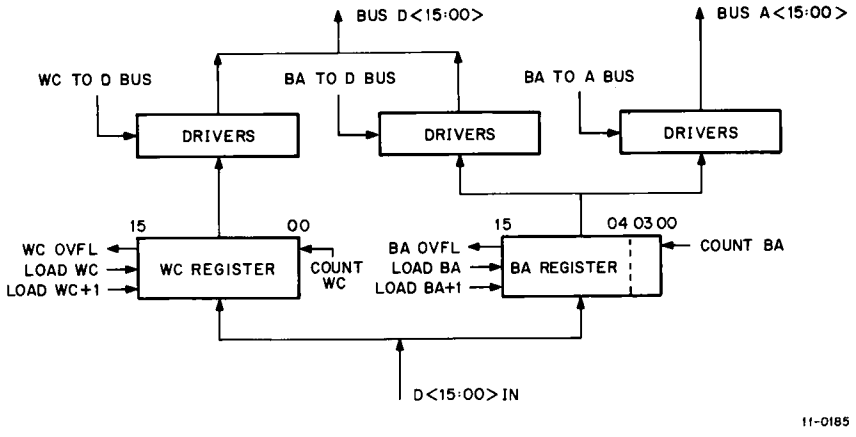


Figure 3-19 M795 Word Count and Bus Address (block diagram)

The BA register can be incremented by either 1 or 2 as a function of a control input (+3V = +1; ground = +2). This incrementation capability allows addressing of either sequential bytes or words. The register is incremented on the trailing edge of a positive pulse applied to the count input of the register. The carry between bits 03 and 04 is broken and brought out to pins on the module. Normally, these pins are jumpered together externally to allow for a full 16-bit count. However, they can be controlled to inhibit the carry and to force repeated addressing of 16 sequential byte addresses. This feature can be used in device-to-device transfers. An overflow pulse is provided as an output whenever the register is incremented from all 1s to all 0s.

The WC register is incremented by either 1 or 2 as a function of its control input. The register increments on the trailing edge of a positive pulse applied to the count input of the register. An overflow pulse is also available. Both registers are reset to all 0s whenever the CLEAR signal is asserted.

A schematic diagram of the M795 module is shown in Figure 3-20. The input signals to the module are listed in Table 3-5, and the output signals are listed in Table 3-6.

Table 3-5
M795 Input Signals

Signal Name	Assertion Level	No. of Signals	Loading	Operation
D(15:00) IN	+3V=1	16	1.5	Data inputs to register.
LOAD WC LOAD WC+1	0V	4	1	Loads data on input into selected byte of register.
LOAD BA LOAD BA+1				Low pulse of 250 ns minimum duration
WC TO D BUS BA TO D BUS BA TO A BUS	0V	3	2	Gates selected register to bus.
CLEAR WC + BA	+3	1	2	Clears all bits. High level of 1 μs minimum duration.
BA INC CONTROL WC INC CONTROL	+3V=+1 0V=+2	2	3	Controls amount of incrementation.

Table 3-5 (Cont)
M795 Input Signals

Signal Name	Assertion Level	No. of Signals	Loading	Operation
COUNT WC COUNT BA	+3V	2	4	Trailing edge of positive pulse increments register (100 ns minimum).
BA CARRY IN	0V	1	3	Carry into upper bits of BA.

Table 3-6
M795 Output Signals

Signal Name	Assertion Level	No. of Signals	Drive Capability	Operation
BA CARRY OUT	0V	1	10	Carry out of low four bits.
BA OVFL WC OVFL	0V	2	10	Register overflow low level pulse.
BUS D⟨15:00⟩	0V=1	16	UNIBUS	Drives data line.
BUS A⟨15:00⟩	0V=1	16	UNIBUS	Drives address.

3.3.7 M796 Unibus Master Control Module

The M796 Unibus Master Control Module provides extremely flexible control logic that is used to control data transfer operations on the Unibus when a device is functioning as bus master. In addition to controlling the four transfer operations (DATI, DATIP, DATO, and DATOB), the M796 module generates strobe and gating signals which transfer both addresses and data to and from the bus; handles deskewing of data received from the bus; protects against data transfers to nonexistent devices by the use of time-out circuits; and provides a flip-flop and integrating one-shot that can be used by the customer for special control functions.

Any device in the PDP-11 system may have the capability of gaining control of the bus and, as bus master, of transferring data to and from other slave devices on the bus. This operation is performed independently of processor control and is usually referred to as Direct Memory Access (DMA). The logic necessary to gain control of the bus is provided by the M782 Interrupt Control Module (paragraph 3.3.5). The M782 module requests use of

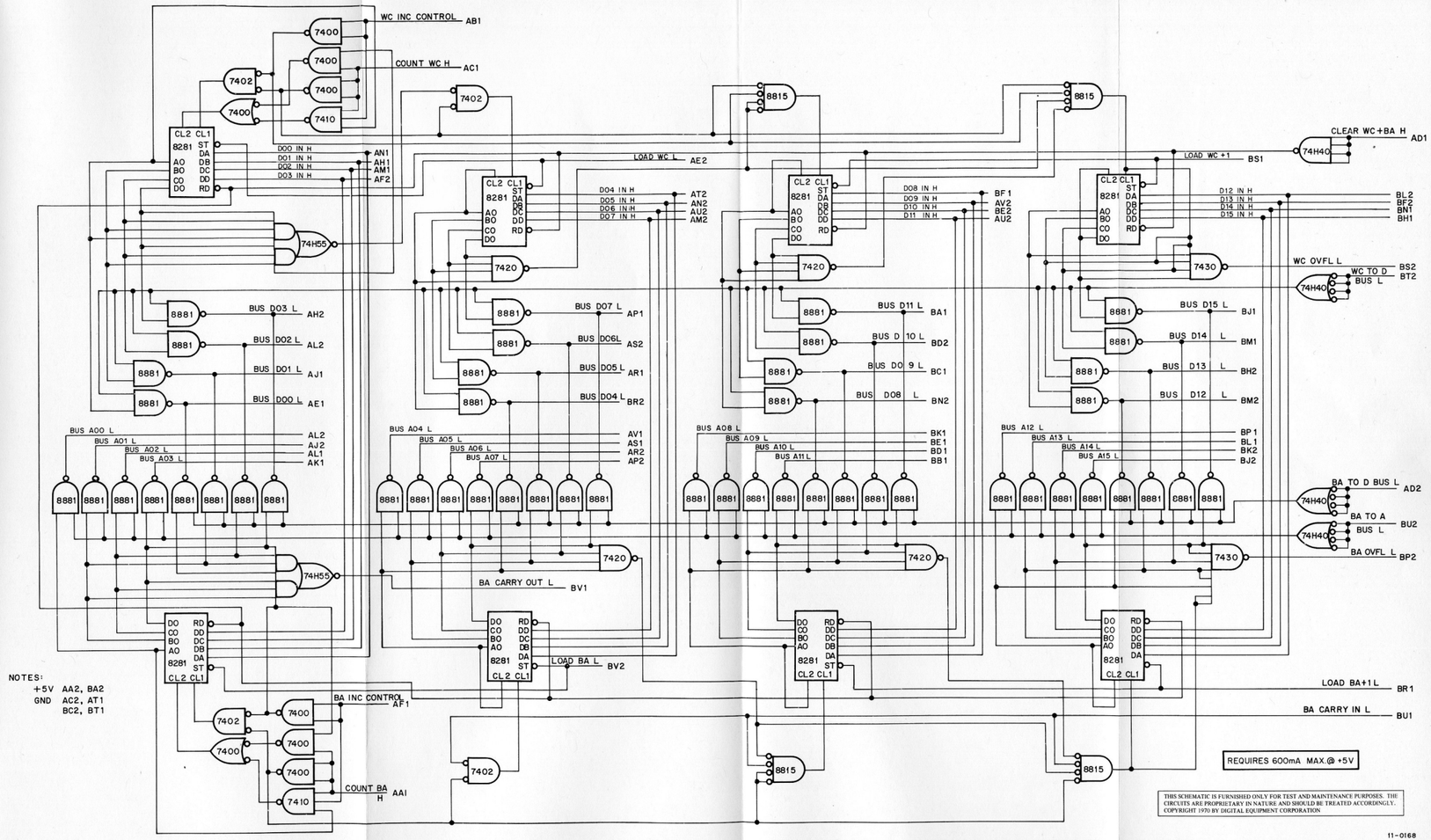


Figure 3-20 M795 Word Count and Bus Address (schematic diagram)

the bus (usually by means of an NPR request), receives the bus grant signal from the processor, asserts selection acknowledge (SACK), waits until the current bus master releases control of the bus, and then asserts BUS BUSY, thereby gaining bus control.

Upon becoming bus master, the device is free to conduct a data transfer. A DATI cycle is performed if the device needs data (either a word or byte) from memory; a DATO cycle is performed if the device is storing a word of data in memory (DATOB cycle for byte storage); a two-cycle DATIP, DATO(B) operation is performed if data held in memory is to be modified as in the case of increment memory or add to memory functions. In order to execute one of these transfer cycles, the device must set BUS C⟨1:0⟩ for the required type of data transfer, specify the address of the slave device participating in the transfer, assert the MSYN signal, and then wait for the SSYN response from the slave. Data must either be gated to D⟨15:00⟩ on a DATO cycle or be received and strobed at the proper time on a DATI cycle. The M796 module performs these functions.

Figure 3-21 is a block diagram of the M796 Unibus Master control module. The BUS C1 and BUS C0 outputs can directly drive the Unibus and are asserted as a function of the control inputs. Table 3-7 lists the states of the control inputs for the four possible bus cycles.

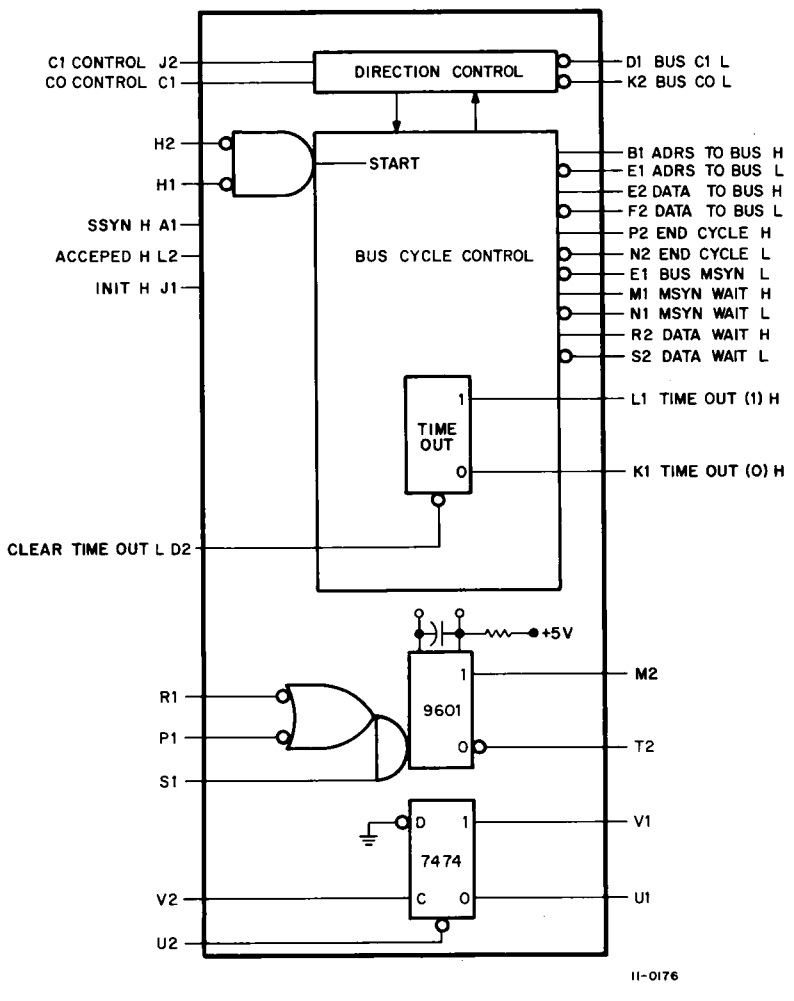


Figure 3-21 M796 Unibus Master Control (block diagram)

Table 3-7
Control Line Input States for M796

C1	C0	Bus Cycle
0	0	DATI
0	1	DATIP
1	0	DATO
1	1	DATOB

NOTE: 0 = low level, 0V
1 = high level, +3V

The data transfer sequence is triggered by meeting the AND condition of two low levels. Usually these two inputs are tied together and are connected to the MASTER signal produced by the M782 Interrupt Control Module. When the AND condition is met, it produces the START signal, which is an internal signal in the M796 module. At the transition of the START signal, both BUS C1 and BUS C0 are asserted as determined by their respective control inputs. The ADRS TO BUS signal is also asserted and is used to gate the address of the slave onto BUS A⟨17:00⟩. If an output cycle is specified (C1 = 1, C0 = 0), the DATA TO BUS signal is asserted and is used to gate the data to be transferred to the slave onto BUS D⟨15:00⟩. The BUS MSYN signal is asserted 200 ns after START becomes true. The master device then waits for a response from the slave.

In a data output cycle (DATO), assertion of SSYN causes BUS MSYN to be negated immediately. After a 100-ns delay, BUS C1, BUS C0, ADRS TO BUS, and DATA TO BUS are negated. When these signals drop, an END CYCLE pulse appears and is usually used to release control of the bus.

In a data input cycle (DATI), the assertion of SSYN produces a 200-ns pulse that appears as DATA WAIT. This delay allows time for the incoming data to deskew and settle. The trailing edge of the DATA WAIT pulse can be used to clock data from the slave into the master device. If a strobe pulse is necessary, the trailing edge of DATA WAIT can be used to trigger the one-shot provided on the module. In either case, once data is received, a positive-going edge is applied to DATA ACCEPTED, causing BUS MSYN to be negated initially, followed by negation of ADRS TO BUS, BUS C1, and BUS C0 100 ns later.

A TIME-OUT flip-flop on the module is set if a SSYN response fails to occur within 20 μs after BUS MSYN is asserted. When this flip-flop is set, the bus cycle is not performed. The TIME-OUT flip-flop is cleared by asserting the CLEAR TIME-OUT signal.

The M796 module provides a special flip-flop that has the clock, reset, 1 side, and 0 side available to the customer. The flip-flop is clocked by a positive transition on the clock input.

An integrating one-shot is also provided on the module. This one-shot is triggered whenever the output of the gating input becomes true: $(\overline{R1} + \overline{P1}) \cdot S1$. The output pulse width of this one-shot is 150 ns but can be lengthened by adding capacitance across the pair of split lugs on the module.

Note that all times mentioned represent nominal values with a tolerance of ±25%. The delays and pulses provided by the module are controlled by simple RC circuits. Therefore, if the customer has any special requirements, part substitutions can be made to alter these time constants.

Figure 3-22 illustrates a typical interconnection schematic for the M796 Unibus Master Control module used in conjunction with the M782 Interrupt Control module. The read/write (R/W) flip-flop is part of the device interface logic and determines the direction of the data transfer (set for a DATO, clear for a DATI). The data transfer is initiated by pulsing SET REQUEST which sets REQUEST BUS. The REQUEST BUS signal generates an NPR

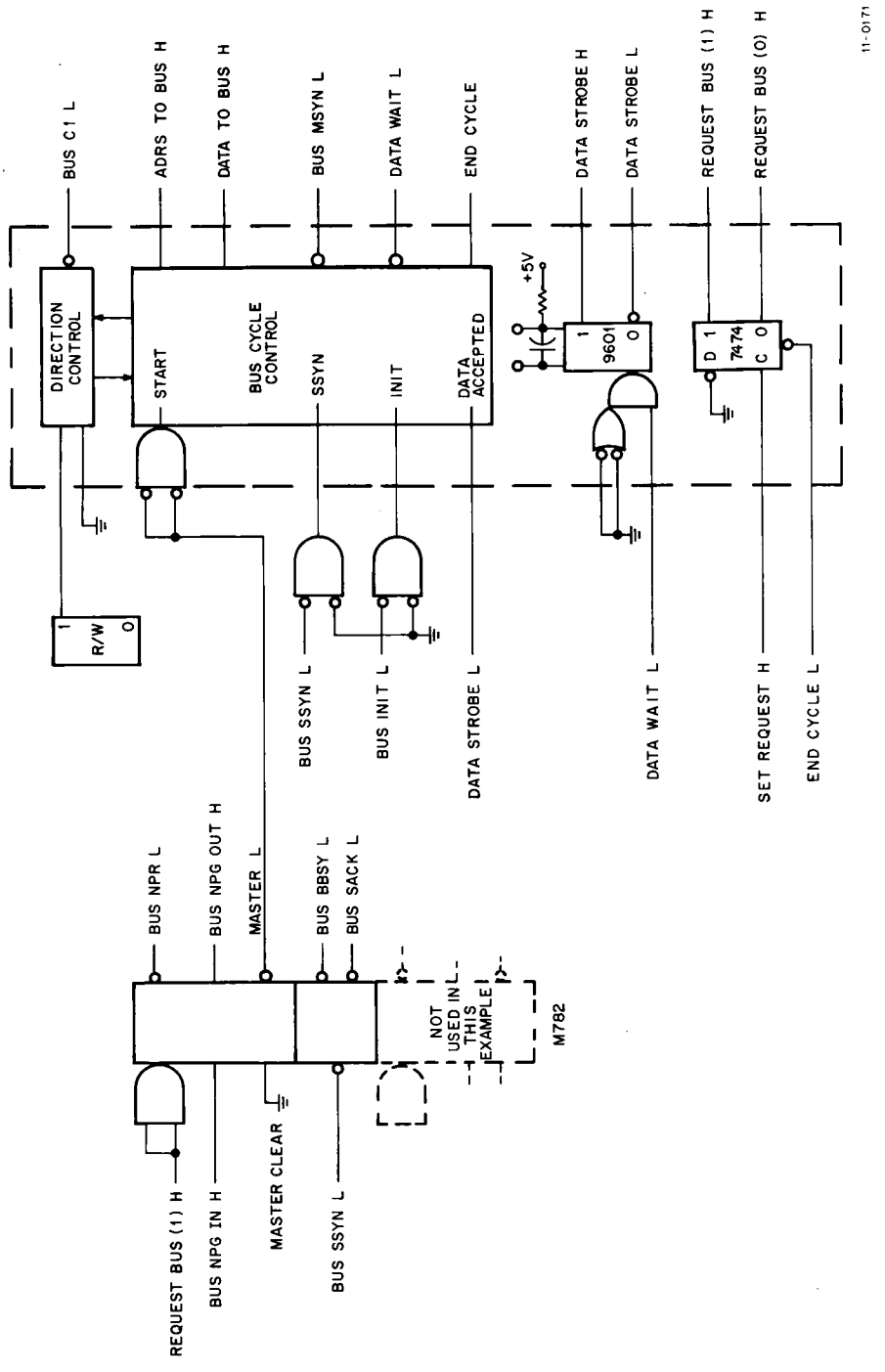


Figure 3-22 M796 Unibus Master Control (typical use)

request which, when granted, gives bus control to the device as indicated by the MASTER signal. The MASTER signal causes the internal START signal to be generated. This signal triggers the sequence of timing signals. Timing diagrams for DATO and DATI cycles are shown in Figures 3-23 and 3-24, respectively.

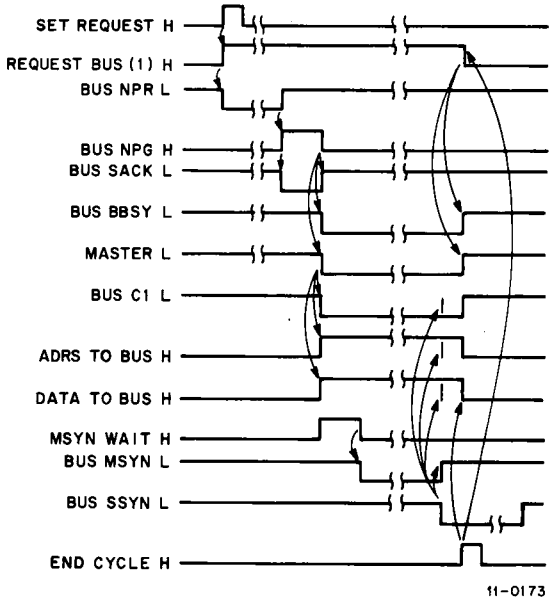


Figure 3-23 M796 Timing Diagram for DATO

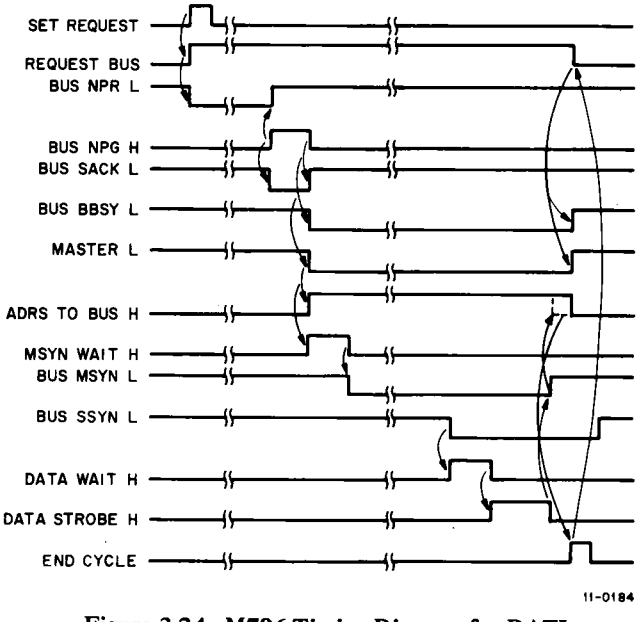


Figure 3-24 M796 Timing Diagram for DATI

Note that in a DATI operation, the DATA WAIT signal is generated when BUS SSYN is received. The trailing edge of DATA WAIT fires the one-shot that produces the DATA STROBE signal. This signal gates the data present on the bus data lines into the device. The trailing edge of DATA STROBE produces a positive transition at the DATA ACCEPTED input that results in the clearing of BUS MSYN.

Additional examples of the use of the M796 module are presented in paragraphs 4.5, 4.7, and 4.10. The M796 circuit schematic is shown in Figure 3-25. The input signals to the module are listed in Table 3-8, and the output signals are listed in Table 3-9.

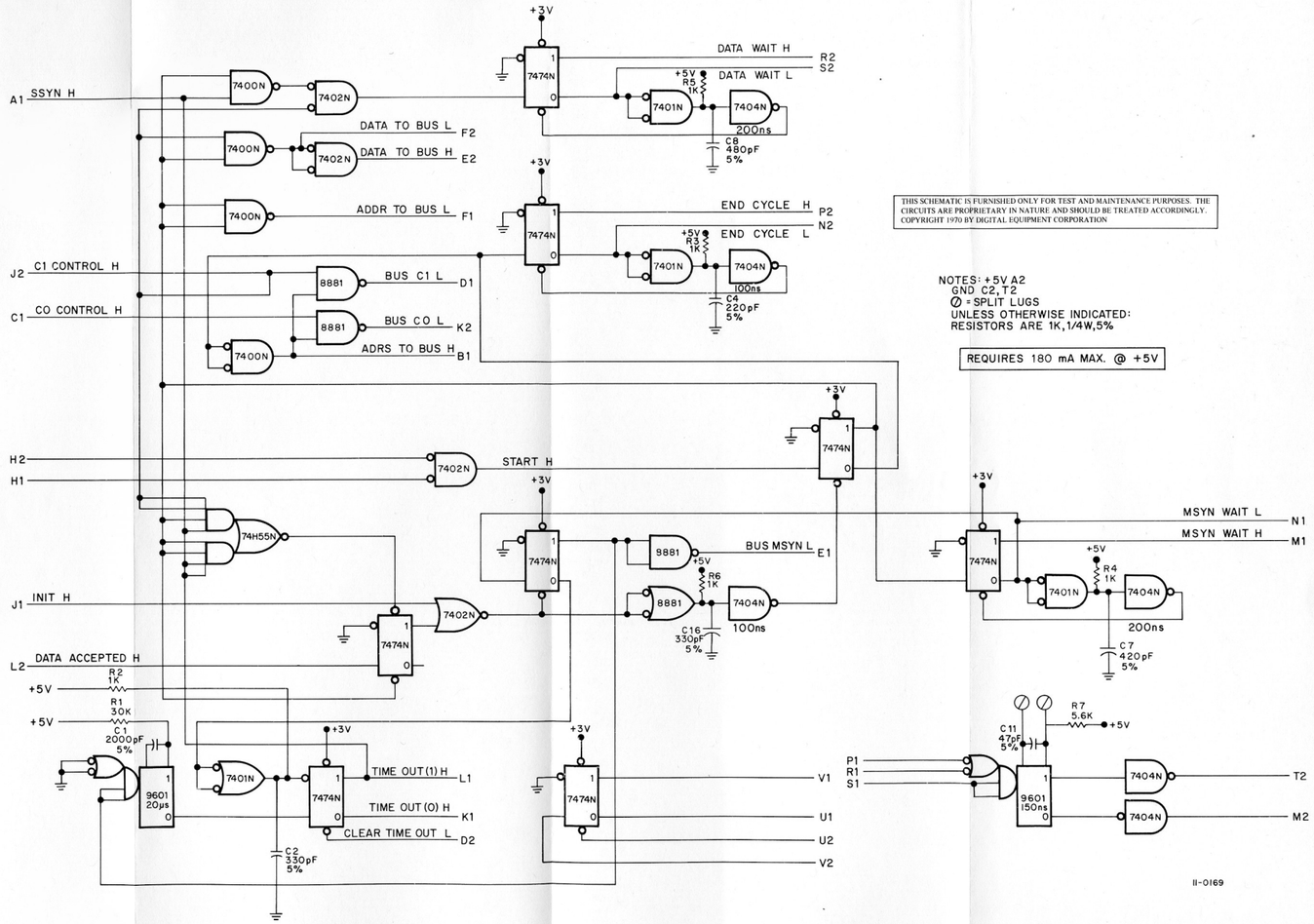


Figure 3-25 M796 Unibus Master Control (schematic diagram)

Table 3-8
M796 Input Signals

Signal Name	Assertion Level	No. of Signals	Loading	Operation
C1 CONTROL		1	5	Controls Bus C1
C0 CONTROL		1	1	Controls Bus C0
PIN H2 PIN H1	L	2	1	Produces START
SSYN	H	1	2	Negates MSYN on DATO
DATA ACCEPTED	H	1	2	Negates MSYN on DATI
INIT	H	1	1	Initializes control
CLEAR TIME OUT	L	1	2	Clears TIME-OUT Flip-Flop
PIN P1 PIN R1	L	2	1	Negative edge triggers one-shot
PIN S1	H	1	2	Positive edge triggers one-shot
PIN V2	H	1	2	Clock input to flip-flop
PIN U2	L	1	2	Clears flip-flop

Table 3-9
M796 Output Signals

Signal Name	Assertion Level	No. of Signals	Drive Capability	Operation
BUS C⟨1:0⟩	L	2	UNIBUS	Drives Unibus control line
ADRS TO BUS	H	1	8	Gates BA to address bus
ADRS TO BUS	L	1	10	
DATA TO BUS	H	1	10	Gates data to bus on DATO
DATA TO BUS	L	1	8	or DATOB
END CYCLE	H	1	10	100 ns-pulse indicating end
END CYCLE	L	1	8	of bus cycle
BUS MSYN	L	1	UNIBUS	Drives Unibus MSYN line
MSYN WAIT	H	1	10	200 ns pulse that delays
MSYN WAIT	L	1	8	assertion of MSYN
DATA WAIT	H	1	10	Allows for deskewing of
DATA WAIT	L	1	8	DATA on DATI. Approximately 200 ns
TIME OUT (1)	H	1	10	1 and 0 side of TIME-
TIME OUT (0)	H	1	10	OUT Flip-Flop
PIN M2	H	1	10	Output of one-shot

Table 3-9 (Cont)
M796 Output Signals

Signal Name	Assertion Level	No. of Signals	Drive Capability	Operation
PIN T2	L	1	10	Output of one-shot
PIN V1	H	1	10	Outputs of flip-flop
PIN U1	L	1	10	

3.3.8 DR11-A General Device Interface

The DR11-A Device Interface (Figure 3-26) is a three-module set that plugs into either a small peripheral slot in the processor or into one of four slots in a DD11 Small Peripheral Mounting Panel. The DR11-A provides the logic and buffer register necessary for transfers of 16-bit input and output data between the PDP-11 System and an external device.

The DR11-A contains three functional sections: a 16-bit buffer register, a 16-bit data input circuit, and a 2-channel flag and interrupt control. These functional elements are shown in Figure 3-27. Address and bit assignments are shown in Figure 3-28. The DR11-A contains three physical modules: an M105 Address Selector, an M782 Interrupt Control and an M786 General Device Interface with two M927 cable connectors.

The 16-bit buffer register is an addressable register that may be read or loaded by instructions transmitted through the Unibus (see Figure 3-28). The register outputs, together with a control signal pulse (NEW DATA READY) used to indicate that the register has been loaded from the Unibus, are available on a printed circuit edge connector #1 which is mounted on the M786 Module. All bits in the buffer register are cleared to 0s by the occurrence of an INIT signal on the Unibus. These signals are logic levels of either +3V (true) or 0V (false). The NEW DATA READY signal is a pulse which has a leading edge coincident with the loading of the buffer flip-flops. The connector accepts an M927 Cable Connector, which contains solder lugs and can be used with ribbon cable, twisted pair cable, or open wire. The M927 is electrically identical to the M904 Module described in the Logic Handbook.

The interface input circuits consist of 16 bus drivers gated to the bus when the input register is read by a DATI bus sequence (see Figure 3-28). The 16 input lines are +3V if true or 0V if false. These signals are also applied to the M786 Module through an M927 Cable Connector and a second printed circuit connector #2. When a DATI sequence occurs, a pulse signal (DATA TRANSMITTED) is applied to the external device.

Two additional request lines are furnished and may be asserted (+3V) by the external device to initiate an interrupt or to generate a flag that may be tested by the program. Whether these two request lines cause an interrupt is determined by two interrupt enable flip-flops which form part of the control and status register in the option (see Figure 3-28). The request lines form two more bits of the status register, independent of the status of the enable flip-flops; thus, they may be tested by the program.

The priority level of both interrupts must be the same, with interrupt A on a higher sublevel than interrupt B. The M786 contains a priority jumper plug which is normally set at BR5. The interrupt enable flip-flops are cleared to 0 (inhibit interrupt) by the occurrence of an INIT signal on the Unibus, or may be set or cleared by the program. Priority may be changed by the jumper plug.

The DR11-A pin assignments are listed in Table 3-10. All inputs are one standard TTL unit load. Inputs have diode protection clamps to ground and +5V. All signals are +3V if true. All outputs are TTL levels capable of

eight unit loads. The NEW DATA READY and DATA TRANSMITTED signals are positive pulses, approximately 2 μs in duration.

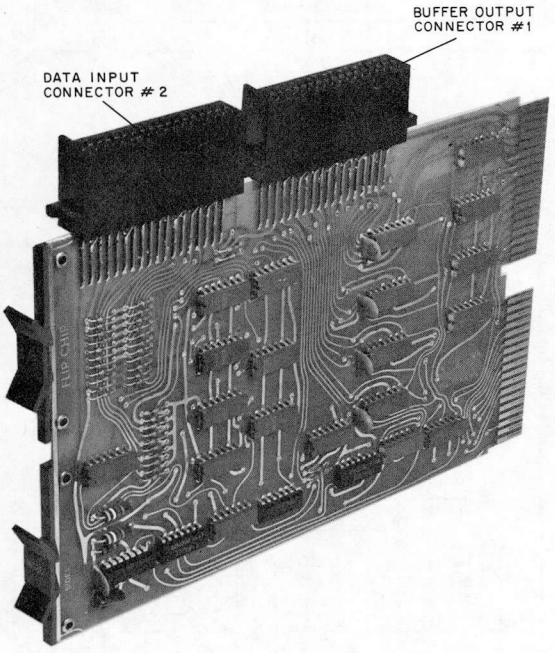


Figure 3-26 DR11-A Device Register Interface

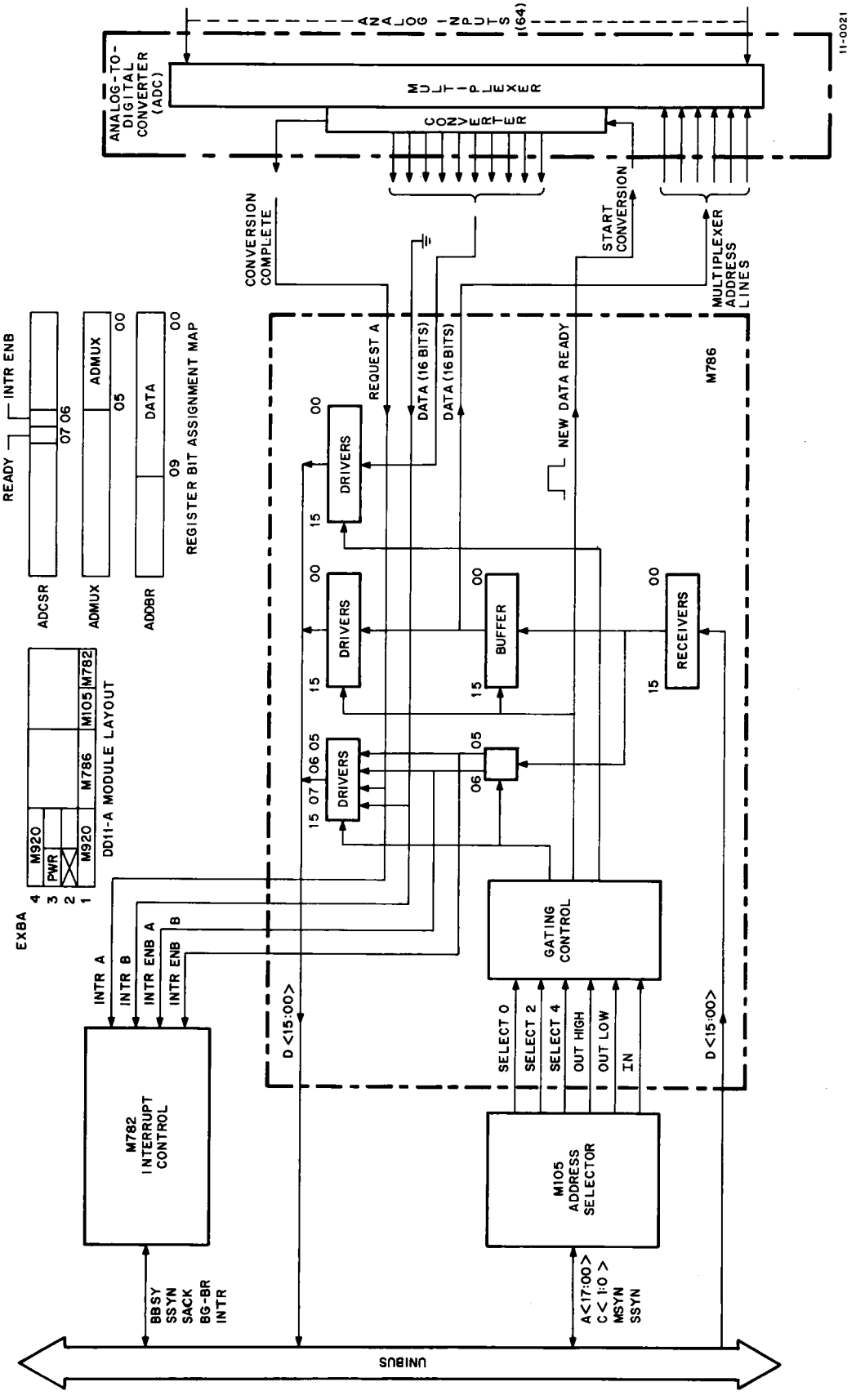


Figure 3-27 DR11-A Device Register Interface (block diagram)

Table 3-10
DR11-A Pin Assignments

Inputs			Outputs		
Signal	Connector	Pin	Signal	Connector	Pin
IN00	2	S1	OUT00	1	P2
IN01	2	S2	OUT01	1	M2
IN02	2	P1	OUT02	1	S1
IN03	2	L1	OUT03	1	P1
IN04	2	P2	OUT04	1	K2
IN05	2	K2	OUT05	1	M1
IN06	2	M1	OUT06	1	S2
IN07	2	T2	OUT07	1	L1
IN08	2	M2	OUT08	1	J1
IN09	2	D2	OUT09	1	H2
IN10	2	E1	OUT10	1	E2
IN11	2	D1	OUT11	1	H1
IN12	2	H1	OUT12	1	D2
IN13	2	E2	OUT13	1	E1
IN14	2	B1	OUT14	1	D1
IN15	2	J1	OUT15	1	B1
REQUEST A	1	T2	NEW DATA RDY	1	V2*
REQUEST B	2	H2	DATA TRANSMITTED	2	V2*

* Pulse signals, approximately 2 μs wide.

Figure 3-29 shows the physical layout of the M786 and M927 modules, and Figure 3-30 is a circuit schematic of the M786 Module. An application of the DR11-A Option is discussed in Section 4.

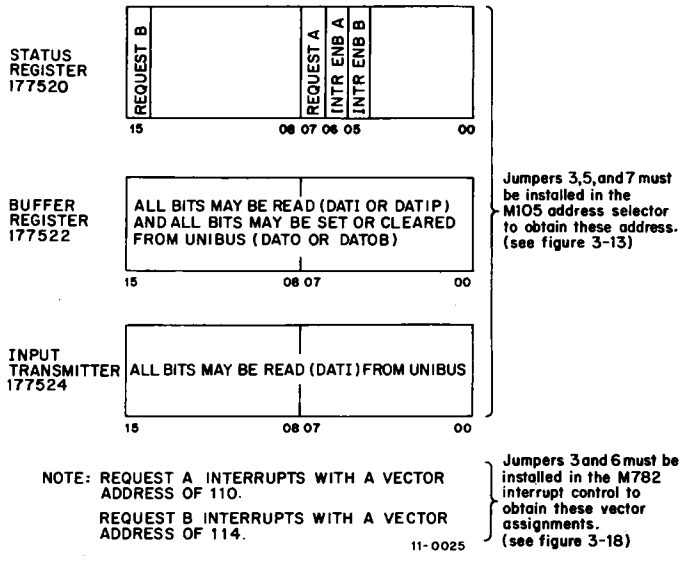

STATUS REGISTER 177520
REQUEST B
REQUEST A
INTR ENB A
INTR ENB B
15
08 07 06 05
00
BUFFER REGISTER 177522
ALL BITS MAY BE READ (DATI OR DATIP) AND ALL BITS MAY BE SET OR CLEARED FROM UNIBUS (DATO OR DATOB)
15
08 07
00
INPUT TRANSMITTER 177524
ALL BITS MAY BE READ (DATI) FROM UNIBUS
15
08 07
00
Jumpers 3,5, and 7 must be installed in the M105 address selector to obtain these address. (see figure 3-13)
NOTE: REQUEST A INTERRUPTS WITH A VECTOR ADDRESS OF 110.
REQUEST B INTERRUPTS WITH A VECTOR ADDRESS OF 114.
11-0025
Jumpers 3 and 6 must be installed in the M782 interrupt control to obtain these vector assignments. (see figure 3-18)

Figure 3-28 DR11-A Address and Bit Assignments

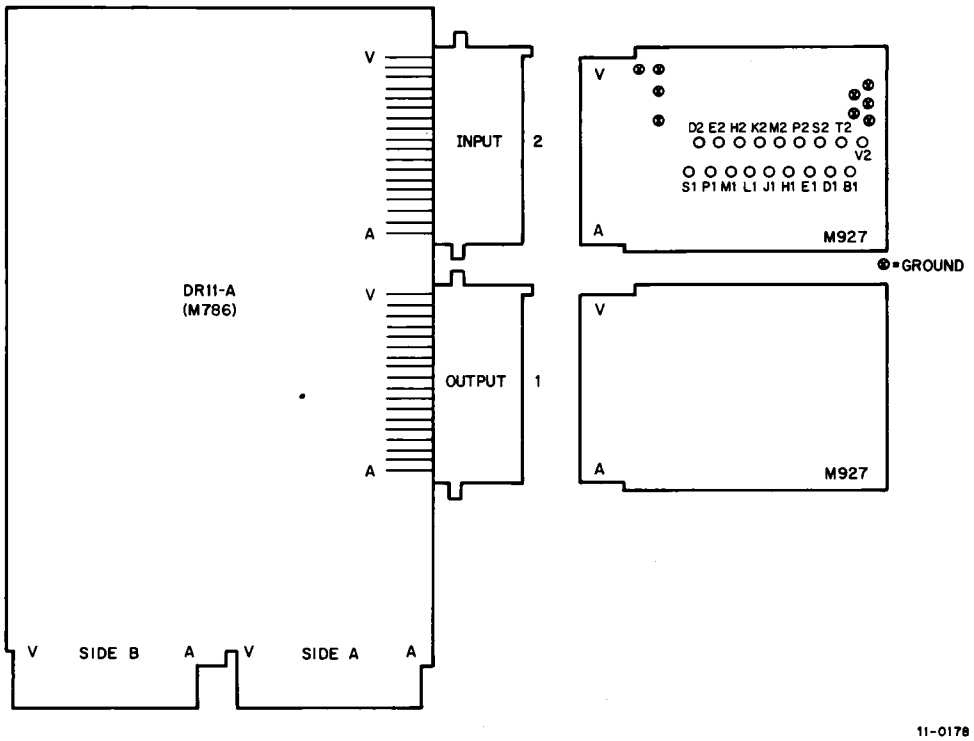
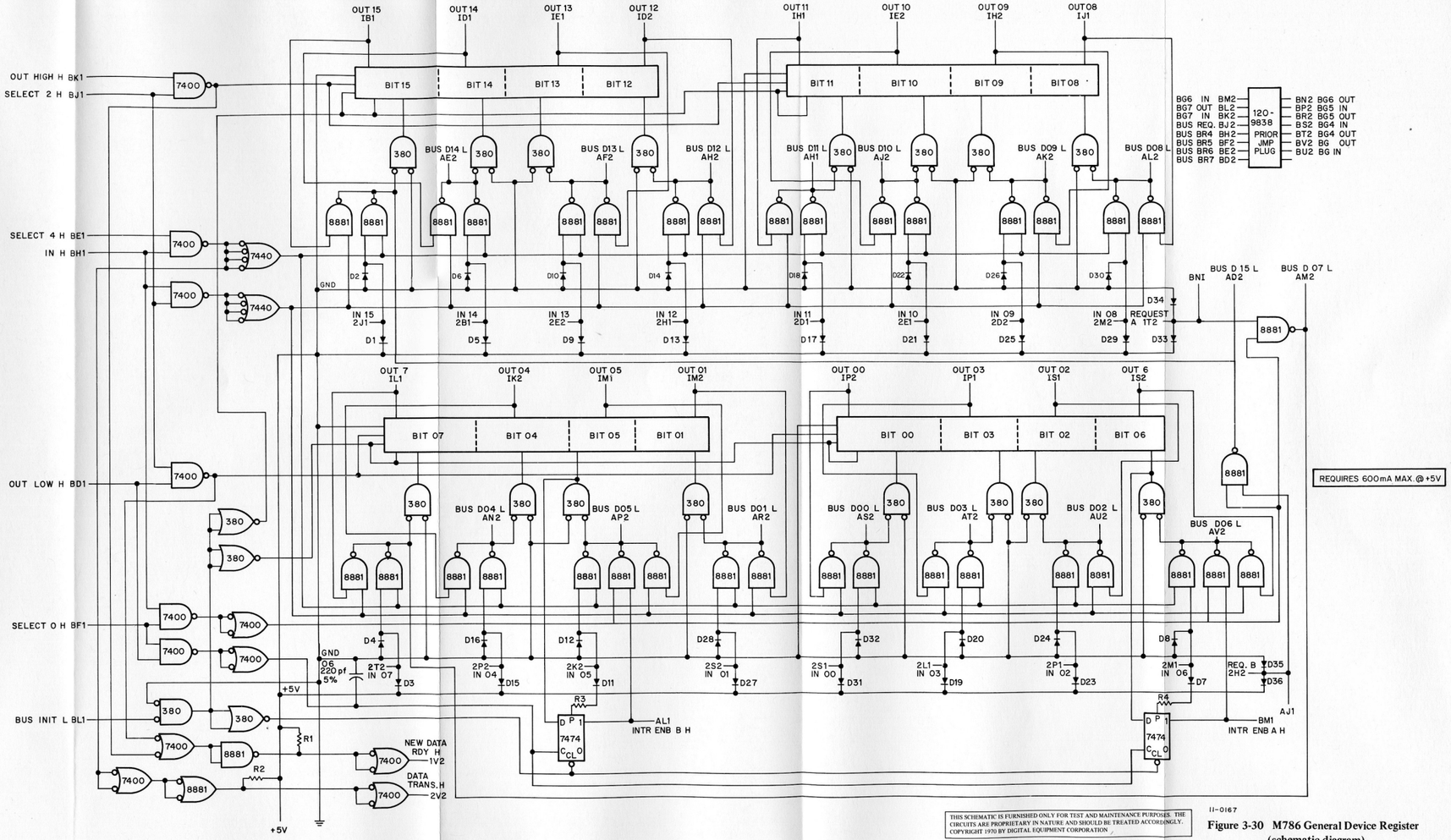


Figure 3-29 DR11-A Physical Layout



UNLESS OTHERWISE INDICATED:
DIODES ARE D664
CAPACITORS ARE .01uf,100V,20%
R1-R2 ARE 390,1/2W,5%
R3-R4 ARE 1K,1/4W,5%
+5V A2
GND C2,T1

THIS SCHEMATIC IS FURNISHED ONLY FOR TEST AND MAINTENANCE PURPOSES. THE CIRCUITS ARE PROPRIETARY IN NATURE AND SHOULD BE TREATED ACCORDINGLY. COPYRIGHT 1970 BY DIGITAL EQUIPMENT CORPORATION

11-0167

Figure 3-30 M786 General Device Register (schematic diagram)

3.4 PDP-11 INTERFACE HARDWARE

The basic design of the PDP-11 System permits highly flexible and easily varied system configurations because of the modular construction. These advantages are available to the user when designing and building custom interface equipment.

The basic building block of the PDP-11 System is the system unit. The system unit is a die-cast zinc frame. Three 288-pin connector blocks are fastened to this frame and each block contains slots for mounting eight single-size Flip-Chip Modules. Peripheral devices and controllers may be made up from one or more of these assemblies. Up to six such system units can be mounted in a BA11 Mounting Box. In addition to space for the system units, the mounting box has space for an H720 Power Supply.

The Unibus is interconnected from system unit to system unit within a single mounting box by M920 Unibus Connectors. A BC11-A Cable Assembly carries the Unibus between mounting boxes.

The basic PDP-11/20 System (see Figure 3-31) occupies four of the six positions within the mounting box. Three of these are system units for the processor, including space for two small peripheral controllers, and one is a system unit for the MM11 Core Memory. System units mounted in the BA11 Mounting Box can accept either the extended-length Flip-Chip modules (8.5 inches long) or standard-length Flip-Chip modules (5 inches long) if equipped with H850 Module Handle Extenders.

Users who prefer to mount the interface equipment directly into a rack can use H911 Mounting Panels and the BC11A Cable Assembly to extend the Unibus. Information on the H911 Panels as well as the M-Series Logic Modules is provided in the *DEC 1970 Logic Handbook.*

The following paragraphs provide a detailed description of available hardware specifically designed for the PDP-11 System as an aid in interfacing and building custom equipment.

3.4.1 BB11 Blank Mounting Panel

The BB11 Blank Mounting Panel (see Figure 3-32) is a prewired system unit designed for general interfacing. The unit contains three 288-pin blocks assembled end-to-end in a casting which can be mounted in either the basic PDP-11 Mounting Box or the extension box. Bus and power connectors, described below, use only 6 of the module slots, thereby leaving 18 slots available for customer use.

The BB11 is wired to accept the Unibus in slots A1 and B1 as shown in Figure 3-33. This connection can be made with an M920 Unibus Connector or a BC11-A Unibus Cable Assembly. All bus signals, including grant signals, are wired directly to corresponding pins in slots A4 and B4. From this point, the Unibus can be continued to the next unit by using an M920 or BC11-A. If the BB11 is the last unit on the bus, slot A4—B4 accepts the M930 Bus Terminator Module. Standard bus pin names are listed in Appendix C.

The bus grant signals are wired through the BB11. These grant signal wires must be removed and replaced with wires to and from the user's control circuits for the grant levels used by the customer-supplied device.

Slot A3 accepts the G772 Power Connector (furnished as part of the BA11 Mounting Box). Power of +5V is distributed to all A2 pins; -15V is distributed to all B2 pins except in slots A1, B1, A4, and B4; and ground is maintained through the frame and power connector on pins C2 and T1 of all slots.

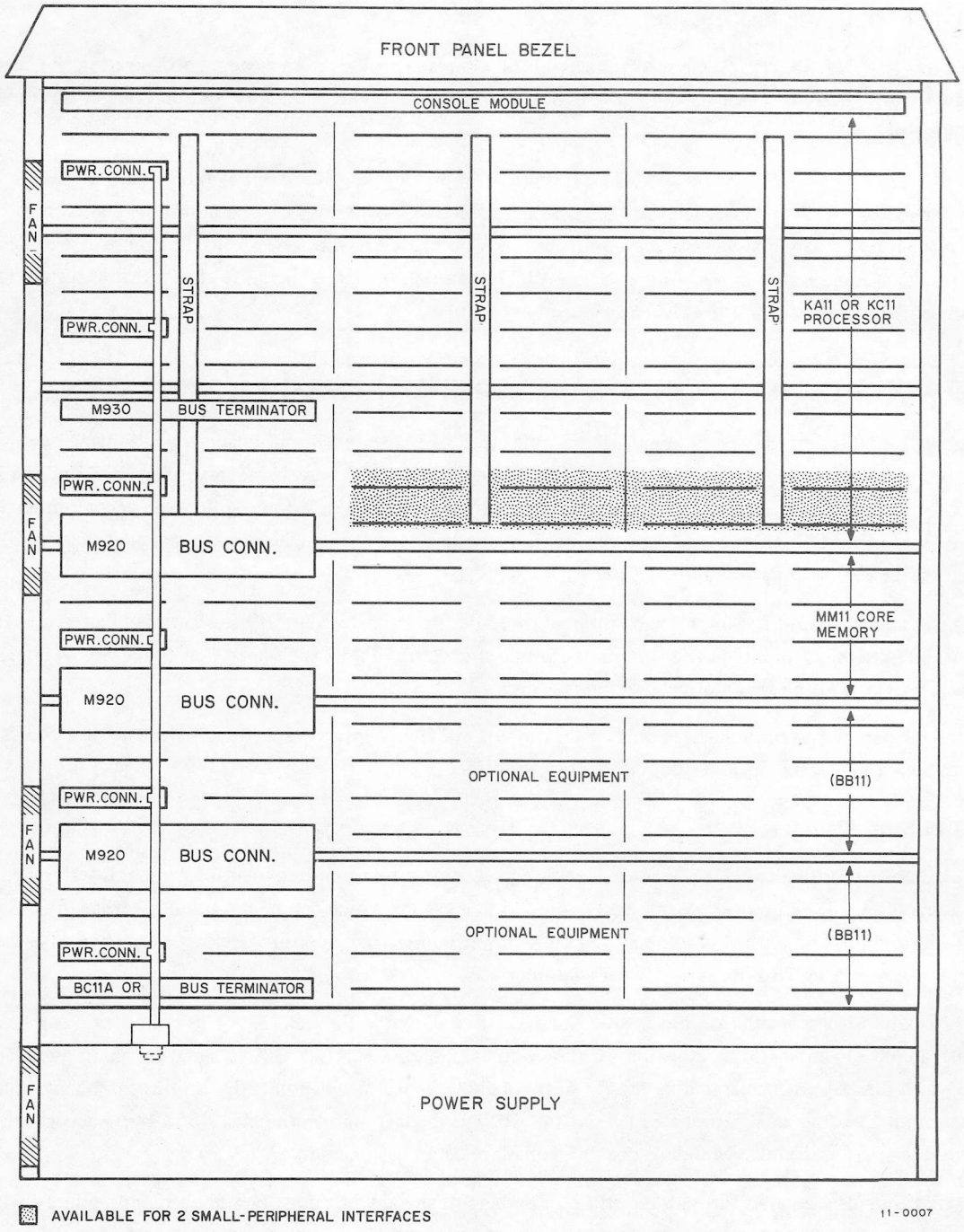


Figure 3-31 Basic PDP-11 Layout with Two BB11 Panels (from module side)

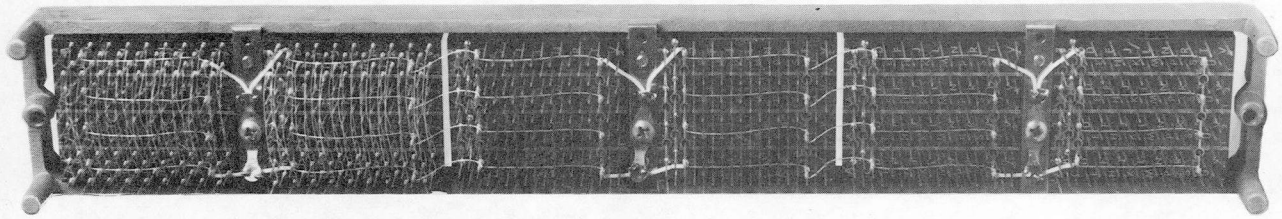


Figure 3-32 BB11 System Unit

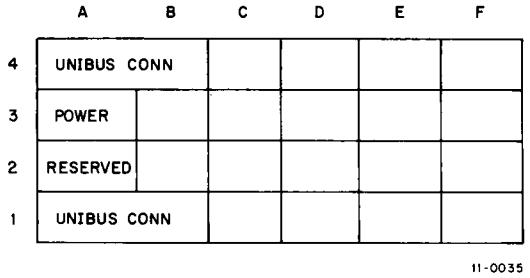


Figure 3-33 BB11 Module Layout

3.4.2 DD11-A Peripheral Mounting Panel

Peripherals may be divided into large and small categories based on the size of the associated interface logic. If the interface logic (including the M105 Address Selector and the M782 Interrupt Control) can fit into four module slots, then the associated device is referred to as a small peripheral. The DD11-A peripheral mounting panel is a pre-wired system unit designed for mounting up to four small peripheral interfaces. The physical construction of the DD11-A is similar to the BB11 blank mounting panel.

Use of the DD11-A requires specialized logic cards for the actual interface, since the pin assignments are fixed for the various control and data signals. Examples of such interfaces are the DR11-A General Interface (M786 module), LP11 Printer Control (M792 module), CR11 Card Reader (M829 module), M792 Diode Memory, and the KL11 Teletype Control (M780 module). Customers may design interfaces to go into a DD11-A by using the W943 module which allows wire-wrapping of the integrated circuit interconnections.

Figure 3-34 illustrates module allocation within a DD11-A unit. Slot A2 must be empty because of power cable overhang; slots B2 and B3 are unused. The DD11-A is wired to permit the installation of four M782 modules in column F, four M105 modules in column E, and four double-height device interface modules in columns C and D. The unidirectional grant lines are wired to column D, and each device interface module must include jumper arrangements to allow selection of the device priority level at the time of installation. Only BR⟨7:4⟩ levels are wired to this column; devices mounted in a DD11-A unit cannot be assigned to the NPR priority level without rewiring the panel. An additional constraint imposed on the interface is that each device may be on only one request level, since both interrupts available through one M782 module are wired in series and must be at the same priority level.

If the device requires interface logic that occupies more than one full row of space in the DD11-A, columns C and D of the second row can be used by using wiring provided between rows 1 and 2 and rows 3 and 4. If this is done, it is no longer necessary to add another M105 and M782 module to slots E and F of the second row.

Additional considerations that must be noted when using a DD11-A as well as specific pin number information is covered in subsequent paragraphs.

3.4.2.1 **Unibus** – The Unibus enters through slots A1–B1 of the DD11-A. This connection can be made with either an M920 Unibus connector or a BC11-A Unibus cable. All bus signals (except grants) are wired directly to corresponding pins in slots A4–B4. Connection can be made with either an M920 connector or BC11-A cable to continue the Unibus to the next unit. If the DD11-A is the last unit on the bus, an M930 Bus Terminator must be placed in slots A4–B4.

3.4.2.2 **Power** – The G772 Power Connector plugs into slot A3. This connector distributes +5V power to all A2 pins and -15V power to all B2 pins except in slots A1, B1, A4, and B4. Ground is maintained through the frame and power connector on pins C2 and T1 of all slots.

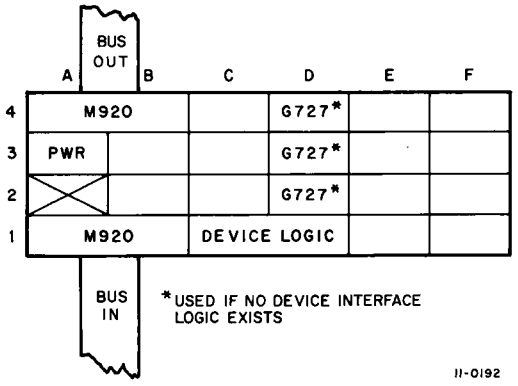


Figure 3-34 DD11-A Module Layout

Note that a G727 Grant Continuity module must always be installed in column D if there is no interface logic in that row. Figure 3-35 shows the wiring assignments that must be adhered to when using the DD11-A.

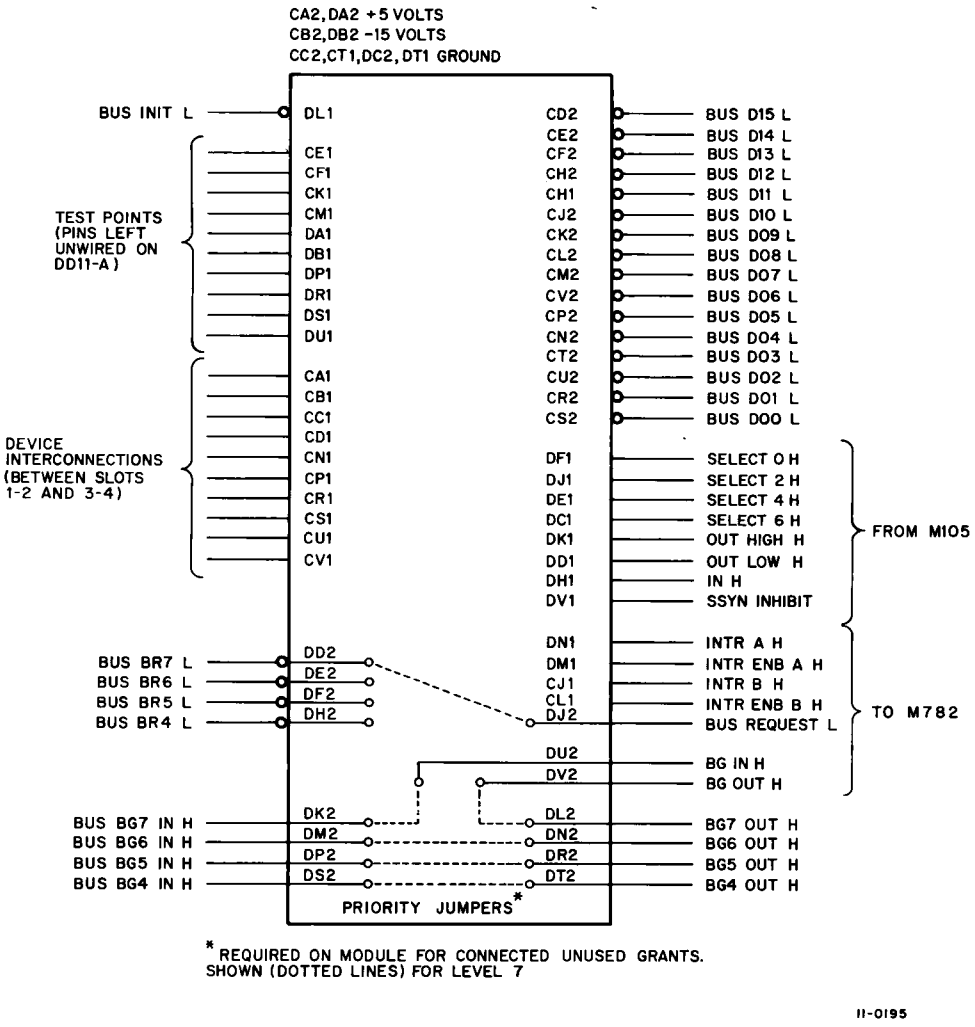


Figure 3-35 Device Control as Wired in the DD11-A

3.4.2.3 **Extended Usage** – Additional wiring provides 10 signal lines between slots C1–D1 and C2–D2 as well as between slots C3–D3 and C4–D4. This permits use of multiple board device controls. Thus, if device logic can be divided into two sections with less than 10 interconnections between sections, then one section can be mounted in slots C1–D1 and the other section in slots C2–D2.

NOTE

Interconnections can be made only between rows 1 and 2 or between rows 3 and 4. No connections can be made between 1 and 3, for example.

Other sections of divided logic can also be placed into slots E and F with the interconnections provided by the normal M105 and M782 module-to-device control signals.

CAUTION

When designing special logic, it is necessary to prevent interference with bus signals prewired to the pins of a particular slot.

3.4.2.4 **Grant Continuity** – The device control module mounted in slots C1–D1 receives the bus grant signal from the Unibus. As a function of its interrupt priority level, this device control must switch the grant signal into its interrupt control (BG IN). After passing through both stages of the interrupt control, the signal (BG OUT) must be returned to the grant chain and passed on to the next device control (mounted in slots C2–D2). In addition, the device control must maintain the continuity of unused grant signals. The BG OUT signals of C1–D1 are wired to the BG IN lines of the next device control. This grant chain must be continued through each device control until the BG OUT signals of the last device control are wired to the outgoing Unibus in slots A4–B4.

Whenever slot D is not used by a device control, a G727 Grant Continuity module must be inserted in this slot. This module provides jumpers between pins K2 and L2, M2 and N2, P2 and R2, and S2 and T2. Three G727 modules and one M920 module are provided with the DD11-A unit.

3.4.2.5 **External Device Cables** – An edge connector mounted on the device control module permits connection to external devices. An H807 36-pin module socket may be mounted on the device control module and an M927 cable connector may be used (refer to paragraph 3.3.8 for a discussion of the M927). The M927 is electrically equivalent to the M908 connector; however, the cable is mounted at the edge of the module card rather than at the end.

3.4.3 BA11 Mounting Boxes

The BA11 Mounting Box (see Figure 3-36) is designed to house the system units that make up a PDP-11 System. The box also includes space for mounting the H720 System Power Supply. The mounting box contains fans for forced air cooling, an insulated top cover (not shown) to prevent debris from falling into the wire-wrap pins (this is necessary because the system units are mounted with the pins up and the modules down), and a foam-lined bottom cover, which serves as a module retainer and minimizes module vibration. The bottom cover serves as part of the air plenum to ensure adequate cooling.

The mounting box is fabricated from zinc-plated steel to resist corrosion. Detailed mounting information, including dimensions, is included in the Installation Planning Section of the PDP-11 Handbook. The four available models are listed in Table 3-11.

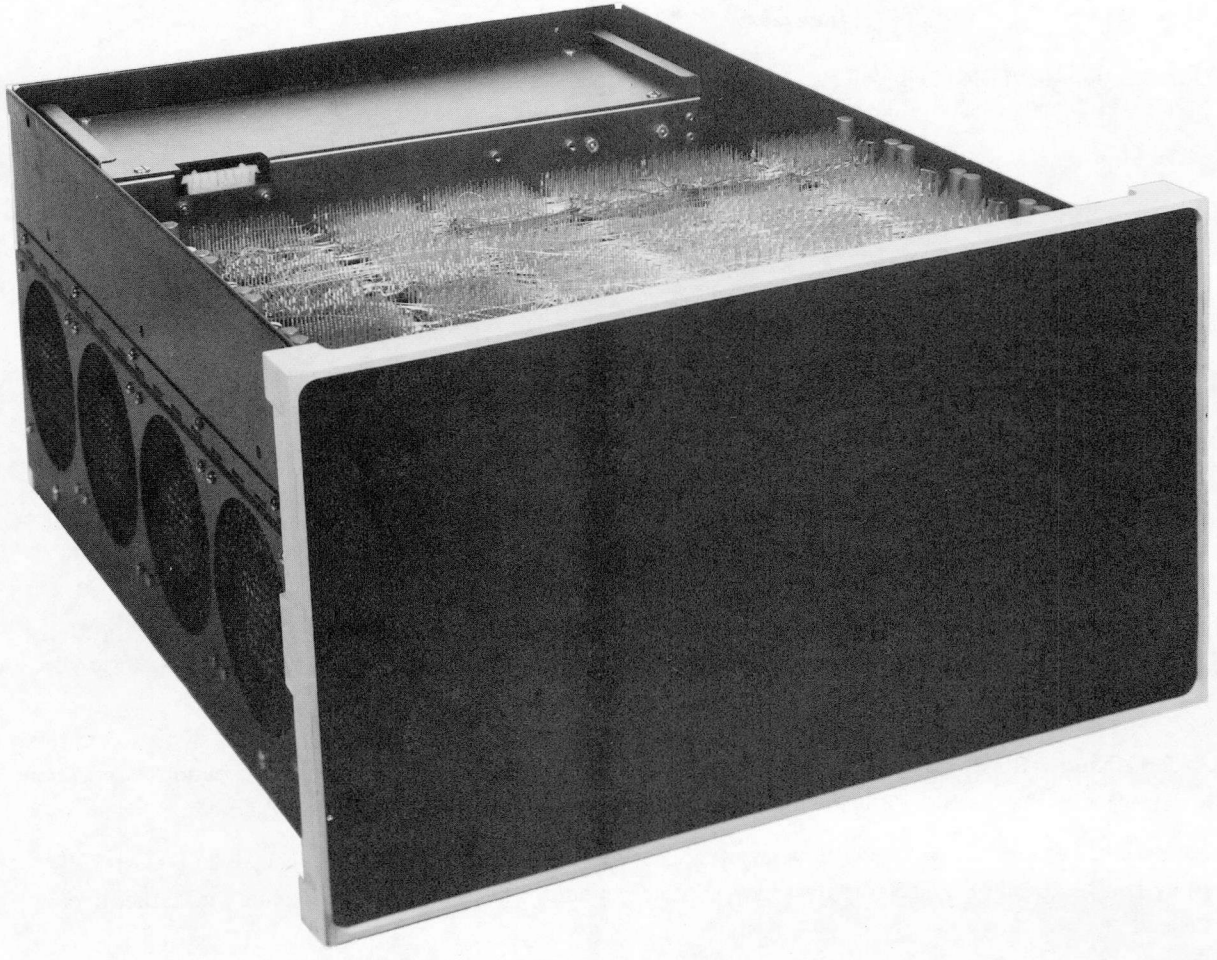


Figure 3-36 BA11 Mounting Box (with wired System Units and H720 Power Supply installed)

Table 3-11
BA11 Mounting Box Models

Model No.	Description
BA11-CS	Basic box*, with tilt-slides for rack mounting
BA11-CC	Basic box, with table-top decorative cabinet
BA11-ES	Extension box**, with tilt-slides
BA11-EC	Extension box, with table-top cabinet

*The basic box includes proper bezel for the console and a key-operated power switch.
**The extension box includes blank front with decorative bezel and does not include a power switch.

3.4.4 H720 Power Supply

The H720 Power Supply is specifically designed to provide power for PDP-11 Systems. It provides +5V and -15V regulated power as well as additional power for console functions. The +5V portion is protected by overvoltage circuits and both portions of the supply feature dynamic current limiting. The basic regulating element is a high efficiency switching regulator that keeps voltages within a ±3 percent tolerance. In addition to providing power, the H720 Supply generates three other signals: a line frequency signal that approximates a squarewave referred to as LTC (line time clock), and the AC LO and DC LO signals previously described in Paragraph 2.2.3. The low voltage detection circuits are interlocked so that -15V is not established until +5V is established. If the +5V fails, the -15V section is shorted to ground to prevent damage to system logic circuits. The power supply outputs are listed in Table 3-12.

There are two H720 Power Supply Models. The H720-A is designed for use with an input of 120V, ±10 percent. The H720-B is designed for an input of 240V, ±10 percent. In addition, the H720-B has taps for lower voltages of 232, 225, and 217, ±10 percent. Both models are designed to operate within a 47- to 63-Hz frequency range. Alternate versions of the H720 (Models C and D) also provide local power control for multibox systems, but have identical dc characteristics.

Table 3-12
H720 Power Supply Outputs

Output	Regulation	Capacity	Remarks
+5V	±5%	16A	
-15V	±3%	12A	
+8V, rms	±15%	1.5A	Full-wave, unfiltered
-24V	±20%	1.0A	Unregulated, filtered
LTC L	0 to 5V		Line time clock
AC LO L	Logic level		ac line low
DC LO L	Logic level		dc line low

CHAPTER 4
INTERFACE EXAMPLES

4.1 INTRODUCTION

Examples of interface designs in Paragraphs 4.2 to 4.10 use the techniques and equipment described in previous chapters. To draw attention to the design features of each interface type, a series of related examples is presented. The first example is a simple basic interface. Each additional example implements several features by adding logic circuits to the previous example. Thus, the first example is the simplest possible read/write interface. This circuit is then used with additional logic to form a program-controlled interface, which in turn is used with additional circuits to form an interrupt-serviced interface, until finally, the circuit is used with additional circuits to form a direct-memory-access interface.

The examples cover input and output transfers and also illustrate techniques for combining the two functions into one interface. Each example includes a description of the operation and logic of the interface, a typical implementation, and programming methods that might be used to operate a device with the interface.

4.2 BASIC INTERFACE

The simplest possible interface, a basic read/write interface, is used when data is transferred to and from the register during bus operations. Applications of read/write, read-only, and write-only registers are discussed in Paragraph 2.7. This particular read/write interface consists of only a storage register and bus gating circuits. The register may be used either as a data register or may be used to drive an output device, such as a set of indicator lights.

4.2.1 Interface Operation

When the basic read/write interface is used, data transfers are under control of the program and the register is assigned an address on the Unibus. During execution of an instruction that addresses the interface, the processor conducts a bus data transfer with the interface register, which responds as a slave. Since a 16-bit register is used, it may be addressed as either a one word register or as two byte (8-bit) registers.

As shown in Figure 4-1, the basic interface uses an M105 Address Selector module to decode the Unibus address lines and to control the clocking of information into the register and the gating of output information from the register to the bus data lines. The register is interfaced to the bus input data lines by ungated receivers, and the inputs are clocked into the register by a strobing signal derived from the M105 Address Selector. The register outputs are gated through the drivers by the GATE REGISTER TO BUS signal. This output gating is necessary to prevent the register from affecting the Unibus data lines when the interface is not participating in a bus data transfer operation.

4.2.2 Data Transfer Operation

The read/write interface can paritcipate in both DATI (or DATIP) and DATO (or DATOB) transfers. Whenever the processor conducts a DATO transfer to the bus address assigned to the read/write register, the data is applied

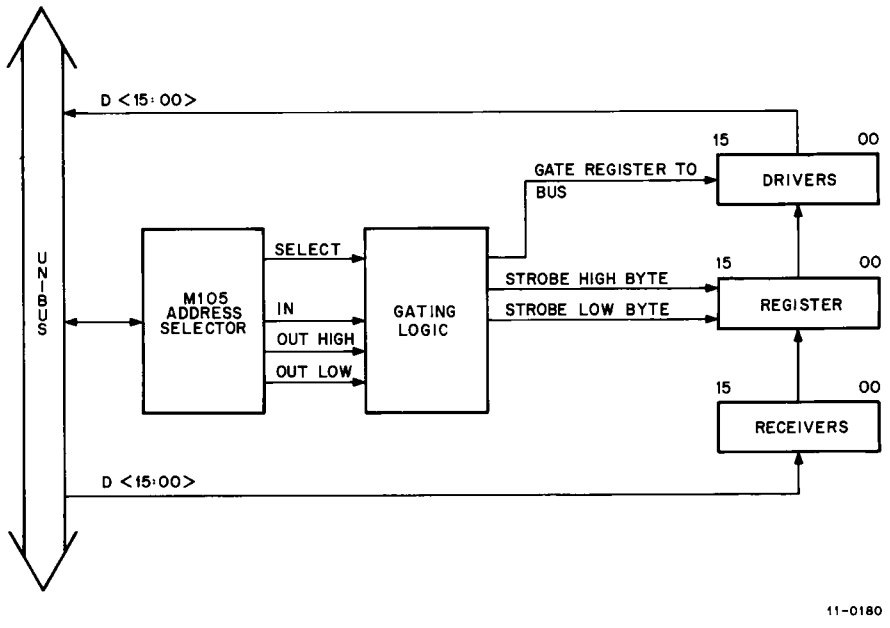


Figure 4-1 Basic Interface (block diagram)

through the bus receivers to the register input. At this time, both the OUT HIGH H and OUT LOW H signals are produced by the M105 Address Selector (see Figure 3-11). When MSYN is asserted by the processor, the decoded address causes the M105 to produce a SELECT 0 signal which is gated by the two OUT H signals to clock data into the register. The Unibus timing guarantees that data is valid 75 ns prior to assertion of MSYN (refer to Paragraph 2.5). Therefore, the inputs have settled before the positive-going transition of the clock signal occurs.

A DATOB transfer functions in a similar manner, except that only one byte of the register is clocked. If address line A00 is 0, the M105 Module asserts OUT LOW H but not OUT HIGH H. If A00 is 1, then only OUT HIGH H is asserted. In either case, data is only strobed into the appropriate byte portion of the register.

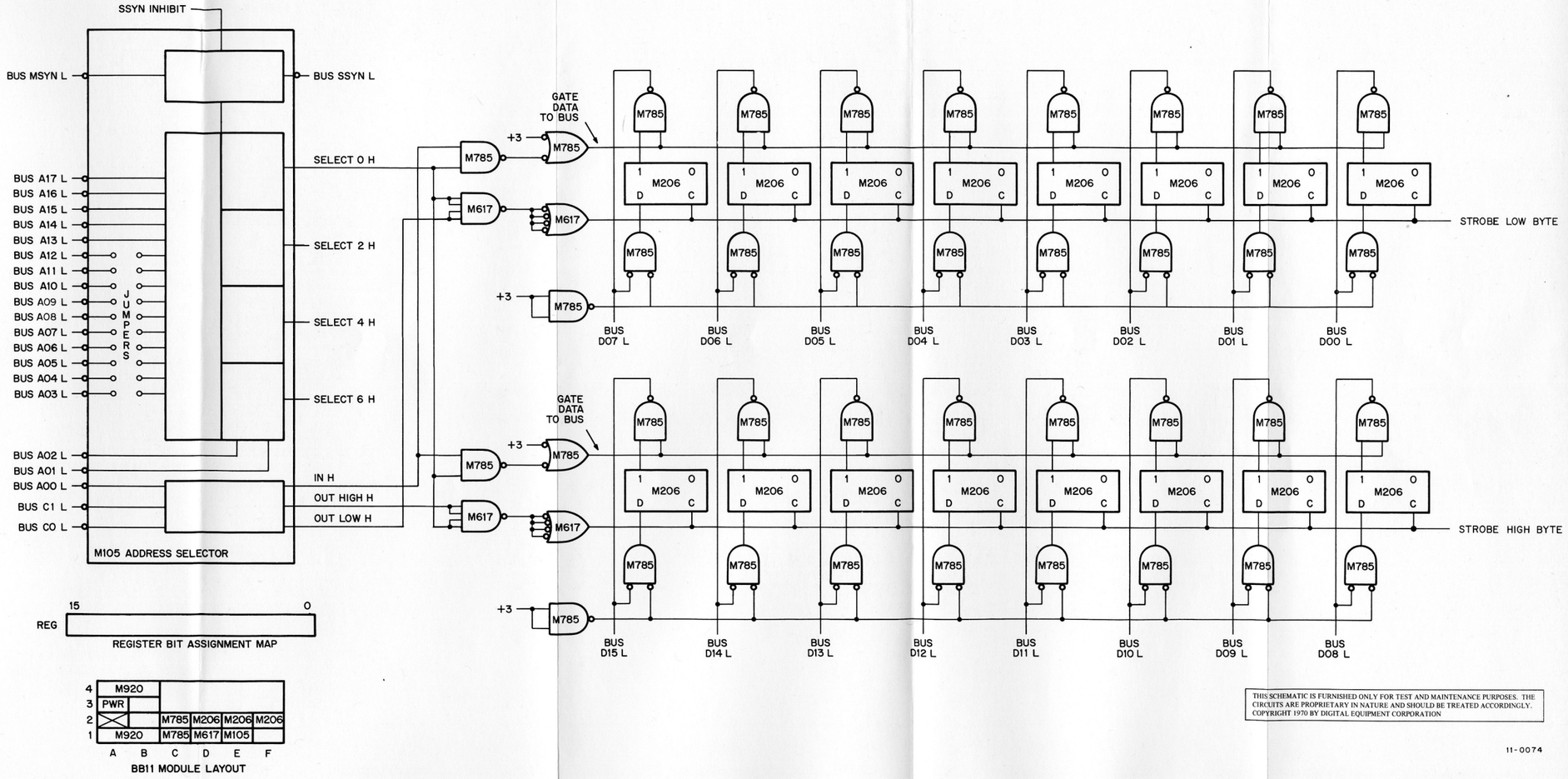
When a DATI transfer occurs, the processor addresses the interface and asserts MSYN. In addition, the M105 Module asserts the same SELECT 0 signal. However, in this case, the SELECT 0 is gated by the IN rather than the OUT signals. The IN signal is generated by the state of the bus C lines. Gating of the SELECT 0 signal by the IN signal produces a GATE DATA TO BUS signal that gates the output data from the register to the Unibus. The M105 Module generates SSYN to indicate that data is ready on the output bus data lines.

CAUTION

In the KA11 and KC11 processors, the destination major state of most instructions consists of DATIP, DATO bus cycles unless the register address mode is used. Appendix B lists all major states and the appropriate bus cycles used by each instruction in the PDP-11 instruction set. When designing interfaces, make certain that the design is compatible with the instruction bus cycles.

4.2.3 Circuit Implementation

Figure 4-2 illustrates a possible method of implementing the circuits in the basic read/write interface. The types and quantities of modules used are: the M105 Address Selector Module (one); the M785 Bus Transceiver Module (two); the M206 General-Purpose Flip-Flop Module (three); and the M617 4-Input Power NAND Gate Module (one). The M105 and M785 Modules are described in Chapter 3 and the M206 and M617 Modules are described



THIS SCHEMATIC IS FURNISHED ONLY FOR TEST AND MAINTENANCE PURPOSES. THE CIRCUITS ARE PROPRIETARY IN NATURE AND SHOULD BE TREATED ACCORDINGLY.
COPYRIGHT 1970 BY DIGITAL EQUIPMENT CORPORATION

11-0074

Figure 4-2 Basic Interface (schematic diagram)

in the *Digital 1970 Logic Handbook*. The modules are interconnected and may be mounted in a BB11 System Unit as shown in the insert on Figure 4-2.

Input data flows from the Unibus, through the input gates on the M785 Modules, to the data inputs of the M206 Flip-Flops comprising the register. The gating provided on the M785 Receivers is not used, and all gates are wired to continuously receive data. Data stored in the register is protected from these changing inputs by the requirement for a clocking signal to load data into the register.

The output data from the register is gated to the bus data lines through the driver sections of the M785 Modules. The M785 Modules are used in this example because the M785 provides the exact combination of input and output gates needed for an 8-bit read/write register. When the number of receivers differs from the number of drivers required in a specific interface, combinations of M783 Bus Driver Modules and M784 Bus Receiver Modules may be used. This example is devoted to illustrating the use and interconnection of bus drivers and receivers rather than indicating the specific modules used in implementation.

4.2.4 Programming the Interface

All data transfers in the basic read/write interface are under processor control, and all memory reference instructions may directly address the interface. If the mnemonic REG is assigned to the register address, the instruction MOV REG, R4 reads the data stored in the register (a DATI operation) and places the data in general register 4 of the processor. The instruction MOV R4, REG reverses the data flow so that the data in general register 4 is placed in the interface register (a DATIP, DATO operation). Any instruction that can access a bus address can conduct data transfers with the interface register. Therefore, the contents of the register may be incremented by an INC REG instruction or summed with an arbitrary value by an ADD VALUE, REG instruction.

4.3 PROGRAMMED DEVICE INTERFACE

A circuit similar to the one in the preceding example is used as the basis for the program controlled interface to an analog-to-digital converter (ADC). It is beyond the scope of this discussion to describe the logic or internal operation of the ADC. The ADC is simply a representative example of many possible external devices that may be interfaced with a design similar to the one discussed in this section. The ADC input and output signals, however, are covered in the following paragraph because of the requirements they place on the interface.

4.3.1 Analog-to-Digital Converter

The analog-to-digital converter used in this example consists of a multiplexer and converter, (see Figure 4-3). The multiplexer selects one of 64 analog inputs and applies it to the converter, which produces the digital equivalent of the analog input.

The interface must provide seven input control signals to the ADC. One input is the start conversion signal, which is a positive transition that causes the ADC to begin the conversion process. The other six control signals are applied to multiplexer address lines so the ADMUX register can be used to select one of the 64 analog inputs.

The interface receives 11 output signals from the ADC. One of these is the conversion complete signal. When the conversion process starts, the conversion complete signal becomes 0V and remains at that level until the conversion is finished. At that time, the signal becomes +3V to indicate that the digital output reflects the analog input (the conversion is complete).

The remaining ten output lines represent the digital value equivalent to the analog input. A zero on any line is indicated by 0V and a one is indicated by +3V.

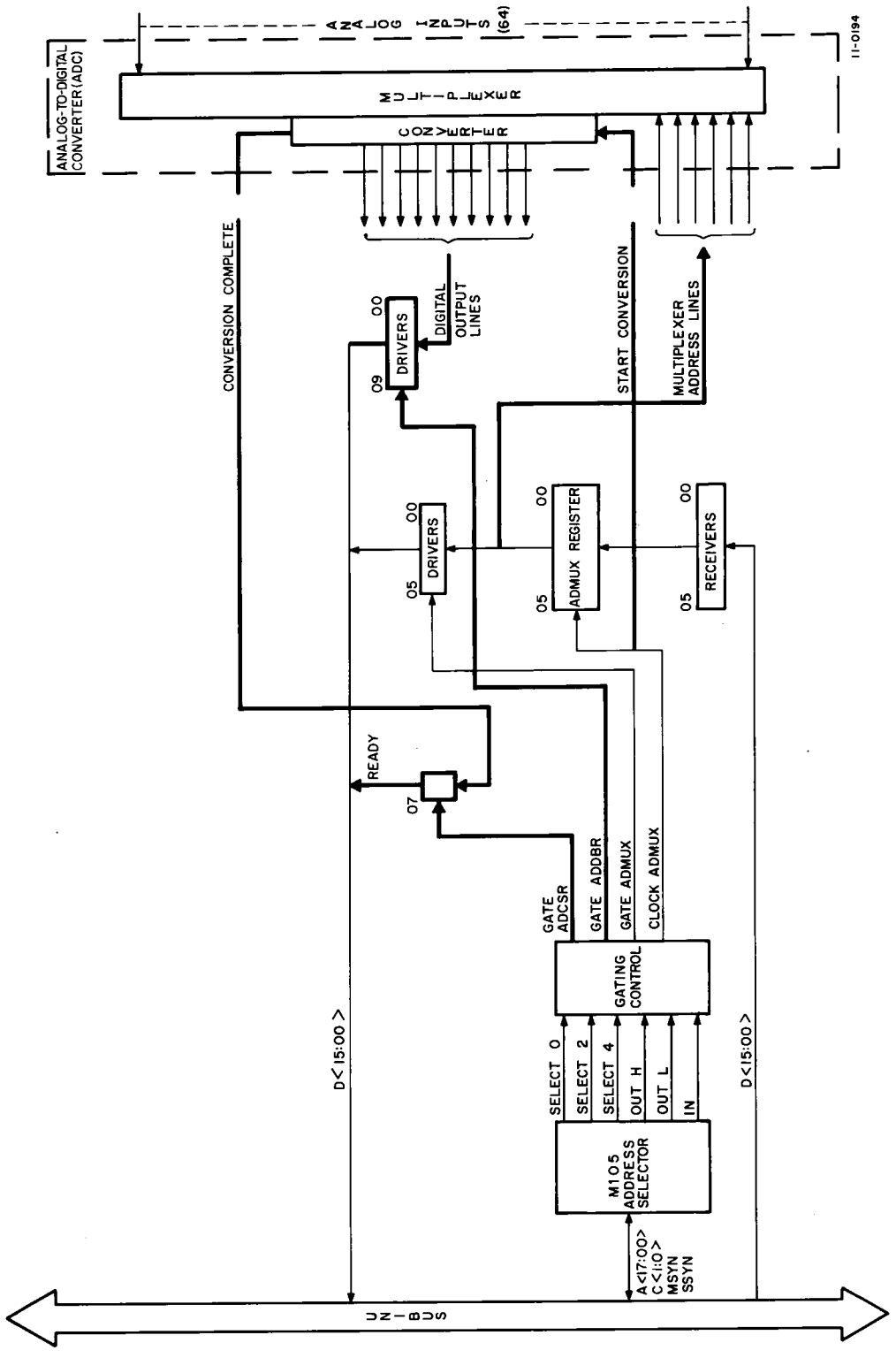


Figure 4-3 Programmed Device Interface (block diagram)

Signal levels used in the interface are standard DEC levels. Both these levels and the signal levels on the Unibus are discussed in Paragraph 3.2.2.

4.3.2 Interface Description

The program controlled interface allows the program to select a specified analog input for application to the ADC and then causes the resultant digitized output and conversion complete signal to be placed on the Unibus data lines to transfer data into the bus master.

The heavy lines in Figure 4-3 indicate logic added to the interface of the previous example. The interface functionally operates with three bus addresses. One address is assigned for the multiplexer (ADMUX) register, which is similar in design to the register in the previous interface example. The second address is for the converted digital output (ADDBR) of a read-only register, and the third address is assigned to a 1-bit control and status register (ADCSR).

The M105 Module decodes the bus address to produce one of three select signals depending on which register is being accessed. The three select signals are gated by IN H and OUT LOW H to produce the four signals (GATE ADCSR, GATE ADDBR, GATE ADMUX and CLOCK ADMUX) shown in Figure 4-3. Only the ADMUX register accepts inputs from the Unibus through the receivers. However, the outputs of all three registers are gated to the bus through separate sets of bus drivers.

Connections between the ADC and interface may be made by a cable connector such as the M908 Module described in the *Digital 1970 Logic Handbook*.

4.3.3 Transfer Operations

The program controlled interface participates in bus data transfers in substantially the same manner as the basic interface described in Paragraph 4.2. Each of the three interface registers can be read during a DATI operation. In addition, the multiplexer (ADMUX) register can be loaded by a DATO operation. Although only the multiplexer register accepts data during a DATO, the other two registers respond when a DATO cycle occurs. If any of the three registers is addressed during a DATO, the M105 Module produces SSYN to complete the bus operation. This is necessary to operate the interface with the processor because the destination operand of all instructions that reference external data (except TST, CMP, and BIT) is transferred by a DATIP, DATO sequence of bus operations. If the interface does not respond to the DATO operation, the processor cannot continue with the program.

4.3.4 Circuit Implementation

Figure 4-4 includes a map of bit assignments for the three registers and a layout for mounting the logic modules in a BB11 System Unit. Neither the M105 Address Selector Module nor the ADC is shown on the figure, but the signals generated by these units are indicated. The connections to the Unibus can be implemented with one M785 Unibus Transceiver Module for the multiplexer register and one M783 Unibus Transmitter Module for the data and control registers. Separate gating must be supplied to use one of the four individual bus drivers on the M783 for a ready bit. The CONVERSION COMPLETE signal is renamed to READY after it passes through the bus transmitter.

4.3.5 Programming the Interface

The START CONVERSION signal, which begins the device cycle, is generated in this interface by the CLOCK ADMUX signal, which loads the multiplexer register. In normal operation, the processor loads the multiplexer register; this action starts the ADC; tests the READY (CONVERSION COMPLETE) bit until the bit is set; and

then transfers the data from the digital output lines of the ADC to the processor. A possible sequence of instructions to perform this task is given below. This program selects an input, waits for the device to complete the conversion, and then transfers the result to register 4.

```
        MOV     INPUT, ADMUX    ;select analog input
READY:  TSTB    ADCSR           ;check for conversion complete
        BPL     READY           ;no, test again
        MOV     ADDBR, R4       ;yes, obtain data
```

INPUT is a location containing the number of the desired analog input line.

A subroutine to examine a series of inputs might be written as follows:

```
MUXSCN: MOV     BUFADR, R4      ;initialize data pointer
        CLR     ADMUX           ;select input line zero
LOOP:   TSTB    ADCSR           ;check for conversion complete
        BPL     LOOP            ;no, test again
        MOV     ADDBR, (R4)+    ;yes, place data in buffer
        CMP     ADMUX, #77      ;last line?
        BEQ     DONE            ;yes, go to done
        INC     ADMUX           ;no, go to next input
        BR      LOOP            ;go to loop
DONE:   RTS     R7              ;exit from subroutine
```

where: BUFADR is a location in core containing the address of the first word of a 64-word buffer
ADCSR is the interface status register
ADMUX is the multiplexer register
ADDBR is the data register

This subroutine is called by the instruction: JSR R7, MUXSCN. The subroutine initializes general register 4 as a pointer to the buffer; initializes the multiplexer register to zero; and sequentially reads the 64 inputs into the corresponding buffer location. When each input has been read once, control returns to the calling program with the contents of general register 4 as the address of the word after the last word of the buffer.

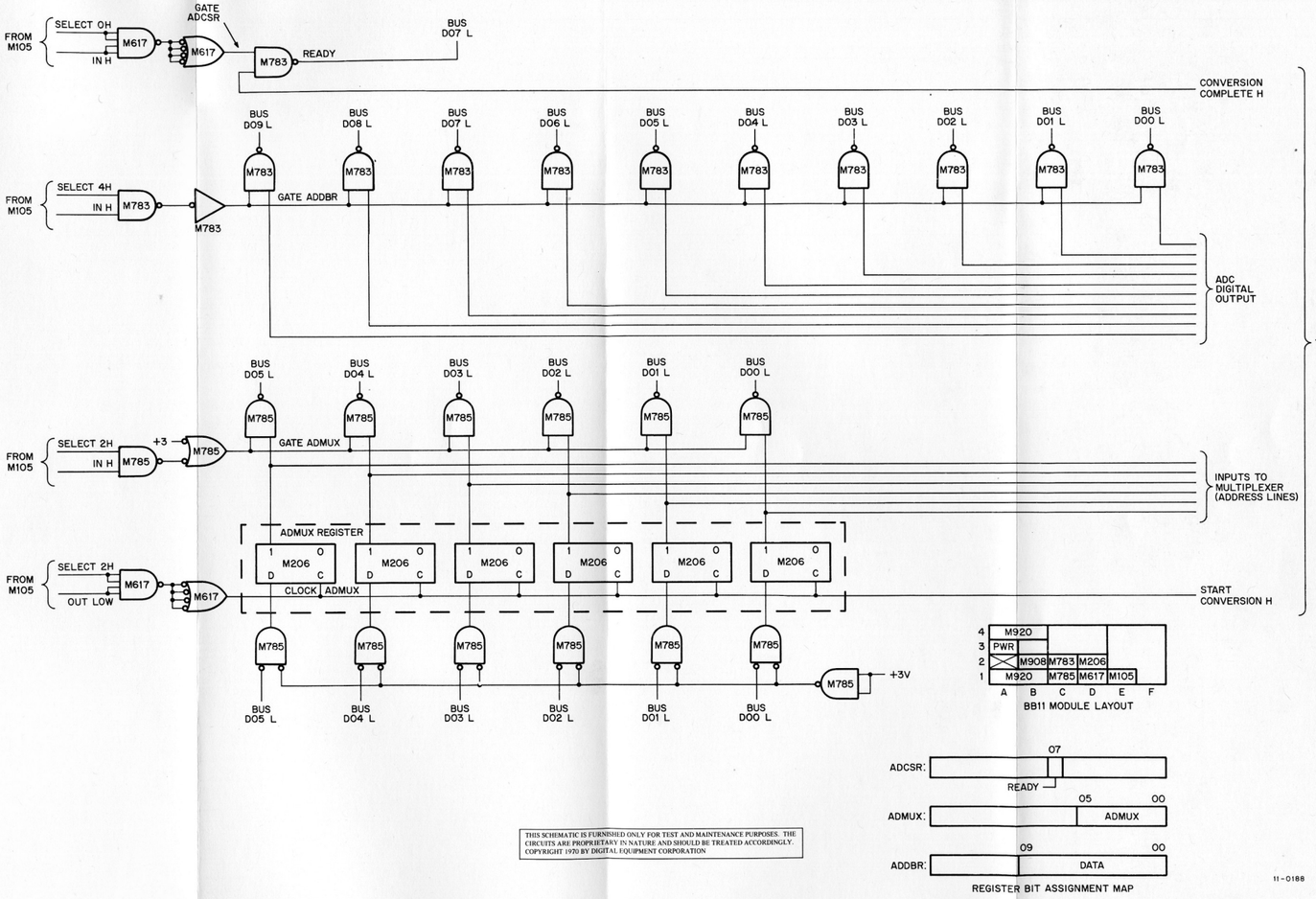
Since loading the multiplexer register starts operation of the device cycle, ADMUX should not be accessed as a destination operand except by a TST, BIT, or CMP instruction. In addition, the INC ADMUX instruction should follow the CMP instruction. This avoids initiating unwanted device operation and allows the subroutine to be immediately re-called.

4.4 INTERRUPT SERVICED INTERFACE

The interface to an analog-to-digital converter would be more versatile if it included an interrupt capability. An interrupt serviced interface with this capability can be formed simply by adding an M782 Interrupt Control Module and one bit to one of the registers in the programmed device interface described in Paragraph 4.3.

The interrupt serviced interface allows the processor to concurrently execute instructions of another program while the analog-to-digital converter (ADC) performs a cycle of operation. The processor responds to a READY (CONVERSION COMPLETE) signal from the ADC by interacting with the device and analyzing the data after it has been collected. This interface eliminates requiring the processor to spend time testing for a ready signal, such as in the case of the programmed device interface.

Whenever a device interface is required, the designer must compare the cost of additional interrupt hardware with the device requirements in terms of transfer speed, frequency of transfers, and amount of use, to determine whether a programmed device interface or interrupt serviced interface is more economical.



THIS SCHEMATIC IS FURNISHED ONLY FOR TEST AND MAINTENANCE PURPOSES. THE CIRCUITS ARE PROPRIETARY IN NATURE AND SHOULD BE TREATED ACCORDINGLY. COPYRIGHT 1970 BY DIGITAL EQUIPMENT CORPORATION

Figure 4-4 Programmed Device Interface (schematic diagram)

4.4.1 Interface Description

Figure 4-5 is a block diagram of the interrupt serviced interface which consists of the programmed device interface with the addition of an M782 Interrupt Control Module, one flip-flop, and one bus driver. This interface can operate either in the same manner as the interface described in Paragraph 4.3 or in an interrupt mode. The additional flip-flop is used to enable or disable interrupt operations. If the flip-flop (which is bit 6 of the control status register) is set by the program, the CONVERSION COMPLETE signal from the ADC causes the M782 Interrupt Control Module to initiate an interrupt. The interrupt operation is described in Paragraph 2.4.2.

4.4.2 Circuit Implementation

Figure 4-6, a circuit schematic of the interrupt serviced interface, also includes a bit assignment map for the three registers and a module layout of a BB11 System Unit. The M782 Module is represented by two blocks similar to the interconnection diagram shown in Figure 3-13. The ADC and the M105 Module are represented by the signal lines connecting them to the logic.

One possible method of implementing the circuit for this interface is to add an M206 General-Purpose Flip-Flop Module and an M782 Interrupt Control Module to the implementation of the programmed device interface described in Paragraph 4.3.4. The M206 Module provides the necessary flip-flop for bit 6 of the control status register, which is used as the interrupt enable bit. The function of the M782 is discussed in the following paragraph. Implementation is illustrated by the module layout in Figure 4-6.

When the ADMUX register is loaded by a DATO or DATOB bus operation, the CLOCK ADMUX signal also starts a cycle of ADC operation. If the interrupt enable flip-flop is set (by loading a one into bit 6 of the control status register by means of a DATO or DATOB operation), then the M782 initiates a bus request when the CONVERSION COMPLETE H (READY) signal is asserted. Although Figure 4-6 indicates that this is a level 5 priority request, any desired bus request (BR) level may be used. When the request is acknowledged, the interface conducts an INTR bus operation. The vector address transmitted to the processor is selected by jumpers on the M782 Module. The program normally responds to the interrupt by collecting the digital output from the ADC and then initiating another conversion cycle. The last conversion cycle is serviced by collecting the data and then transferring to an evaluation routine.

4.4.3 DR11-A Implementation

A more convenient method of implementing an interrupt serviced interface is to use an M786 General Interface Module to make a DR11-A 16-Bit General Interface. Figure 4-7, a layout of the modules mounted in a DD11-A System Unit, shows the savings in space and interconnections compared to the implementation method discussed in Paragraph 4.4.2. The DD11-A System Unit is prewired to accept four DEC small peripheral interfaces; e.g., DR11-A. A discussion of the DR11-A, including specifications, is presented in Paragraph 3.3.6.

Figure 4-7 is similar to Figure 4-5 because the DR11-A logic is used in the same manner and with the same programs as any other logic used to implement an interrupt serviced interface. The M786 Module portion of the DR11-A provides cable connectors; therefore, no additional wiring or connectors are required.

Connections between the ADC and the M786 Module are made as follows:

CONNECTOR	DR11-A	ADC
1	OUT (06:00)	Multiplexer inputs
	NEW DATA READY	Start conversion
2	IN (09:00)	Digital outputs
	REQUEST B	Conversion complete

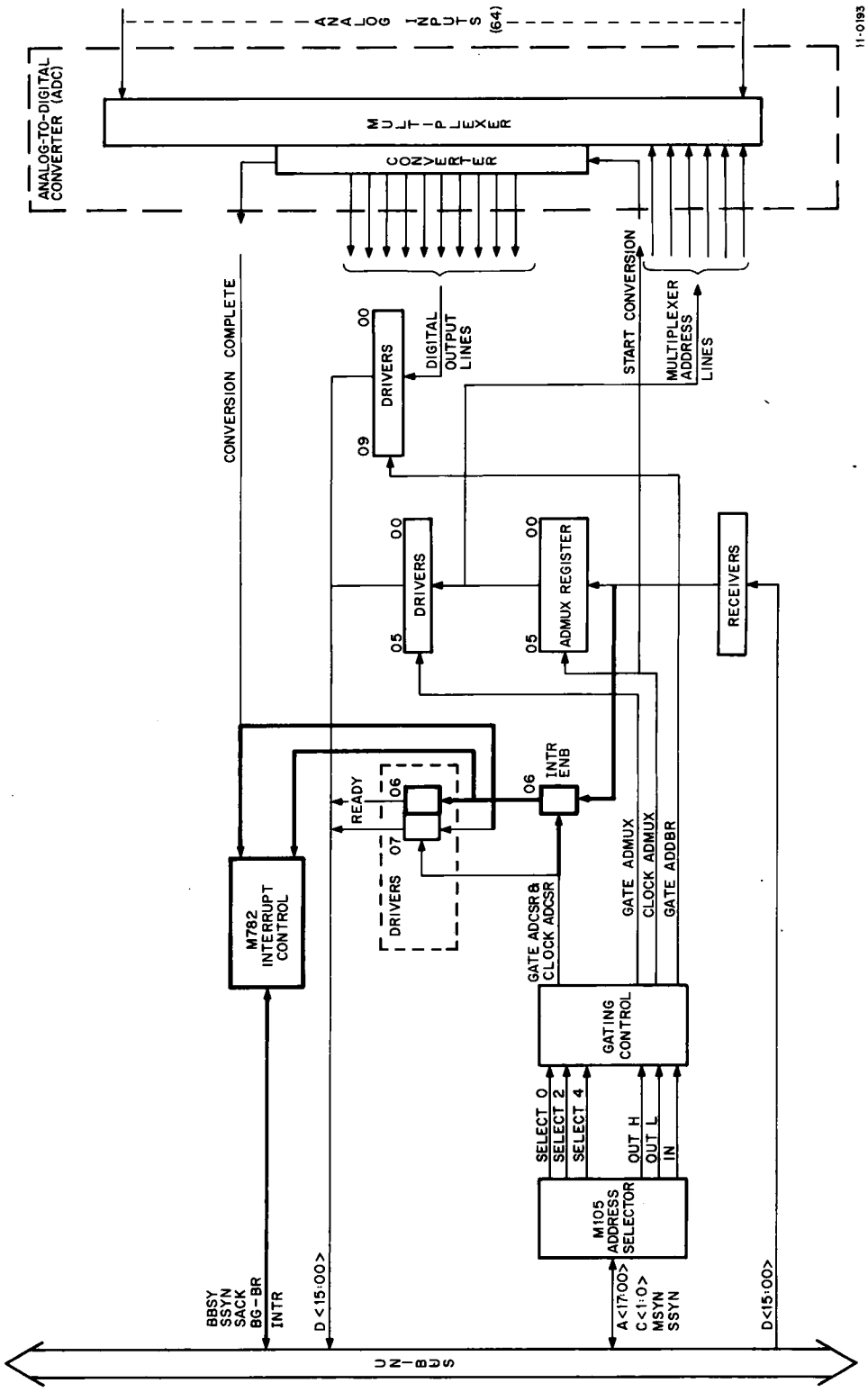
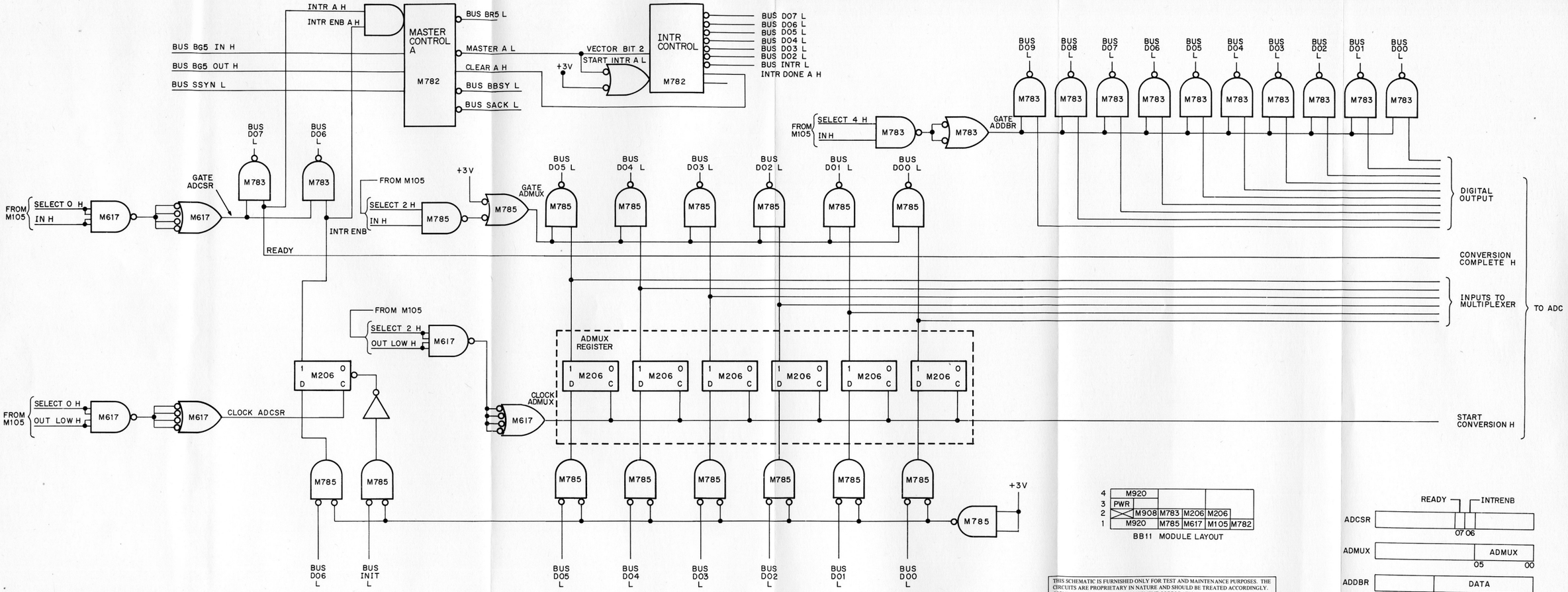


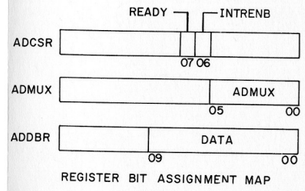
Figure 4-5 Interrupt Serviced Interface (block diagram)



4	M920					
3	PWR					
2		M908	M783	M206	M206	
1	M920		M785	M617	M105	M782

BB11 MODULE LAYOUT

THIS SCHEMATIC IS FURNISHED ONLY FOR TEST AND MAINTENANCE PURPOSES. THE CIRCUITS ARE PROPRIETARY IN NATURE AND SHOULD BE TREATED ACCORDINGLY. COPYRIGHT 1970 BY DIGITAL EQUIPMENT CORPORATION



REGISTER BIT ASSIGNMENT MAP

11-0038

Figure 4-6 Interrupt Serviced Interface (schematic diagram)

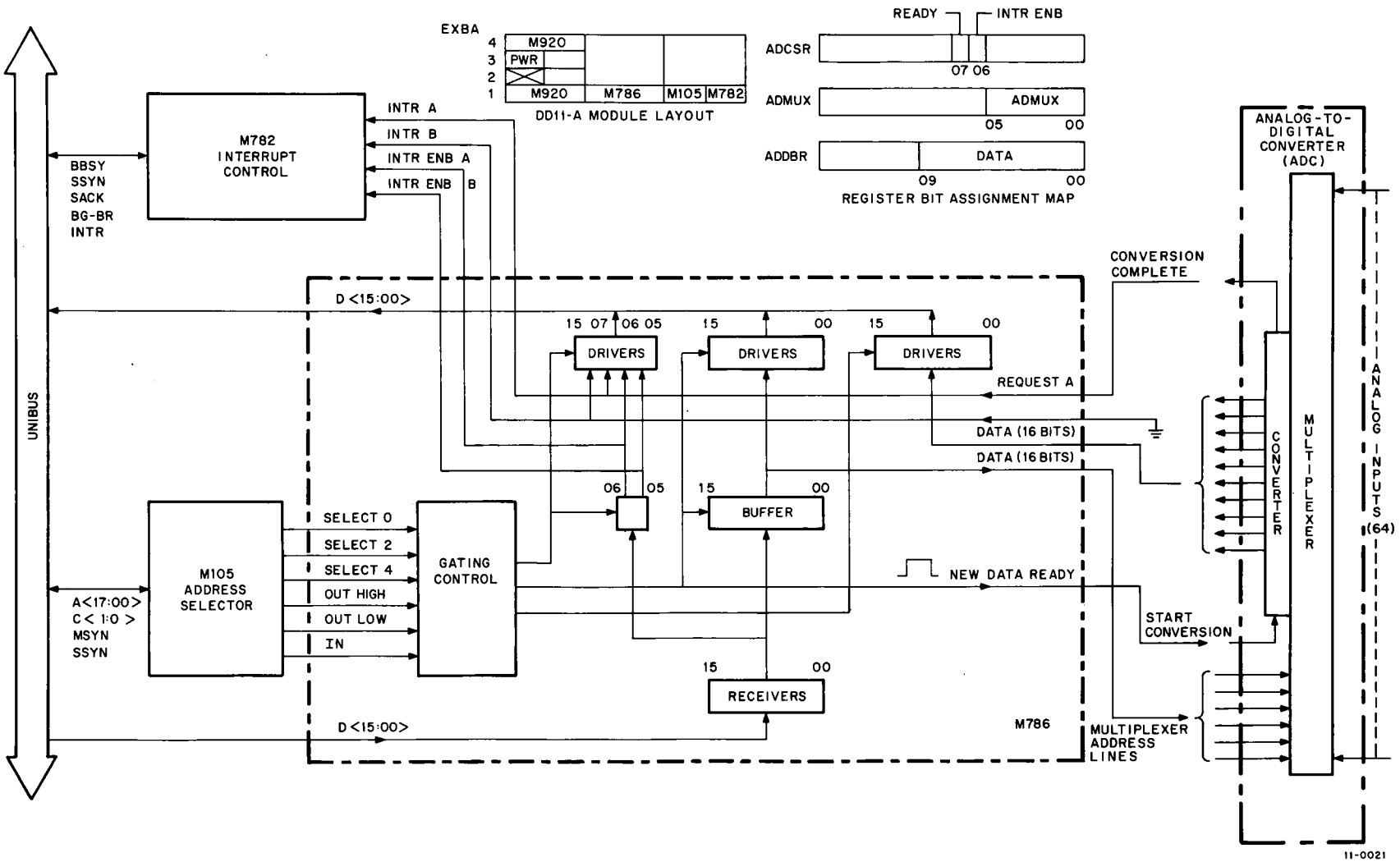


Figure 4-7 DR11-A Implementation (block diagram)

4.4.4 Interface Programming

The following program is a typical interrupt service routine that collects data from the ADC and enters an evaluation routine after the final conversion cycle.

```
ADCVEC:   ADCSER                  ;set up ADC vector area
          240                     ;status includes priority level 5
            .
            .
            .                     ;main program follows
BEGIN:    MOV BUFSTRT, BUFADR     ;initialize buffer pointer
          CLR ADMUX               ;start multiplexer at channel 0
          MOV #100, ADCSR         ;enable interrupt
            .
            .
ADCSER:   MOV ADDBR, @BUFADR      ;collect data
          CMP BUFADR, BUFSTRT+174 ;last one?
          BEQ DONE                ;yes, go to DONE
          ADD #2, BUFADR          ;no, increment pointer
          INC ADMUX               ;increment multiplexer and
                                  ;start conversion
          RTI                     ;return to main line
DONE:     CLR ADCSR               ;clear interrupt enable
            .
            .                     ;follow this with the
                                  ;evaluation routines
```

where: ADCSR, ADMUX, and ADDBR are the device registers in the interface
BUFSTRT contains the starting address of a buffer
BUFADR is a location to be used by the device service routine
ADCVEC is the address specified by jumpers on the M782 Module and contains the address of the device service routine tagged ADCSER
ADCSER device service routine

After the initiation instructions in the main program are executed, the interrupts cause the processor to execute the ADCSER routine. The last time this is performed, the evaluation routine is also executed.

The CLR ADMUX instruction should precede the MOV #100, ADCSR instruction to prevent the interface from causing an immediate interrupt, which could occur if the interrupt enable bit is set when the device has the conversion complete (ready) signal asserted.

For longer service routines, in which the use of registers is desirable to increase access speed to certain words, the previous register contents could be pushed onto the processor stack at the beginning of the routine and popped back in the general register before returning to the main program. Registers should not be used in interrupt service routines without this procedure because the use of the general registers by the main program is not predictable. The service routine is not much shorter than the programmed device routine in the previous example; however, it does have the major advantage of not tying up the processor while the device is performing an operation cycle.

If the evaluation routine is to return control to the main program at the same point in the service routine, this may be accomplished by terminating the evaluation routine with an RTI instruction. If any other type of return is used, the program must remove the old PC and PS that were placed on the stack by the interrupt operation. Removal is accomplished by executing an ADD #4, R6 instruction.

4.5 DIRECT MEMORY ACCESS (DMA) INTERFACE

The direct memory access (DMA) interface conducts data transfer operations to place data from the device directly into memory. A DMA interface performs a large number of transfers without any processor intervention, thereby reducing program and execution time overhead. After the interface device registers are initialized, all transfers take place under control of the interface, thereby eliminating processor time. The processor is notified by an interrupt when all the data has been transferred and the program responds appropriately.

Figure 4-8 is a block diagram of a DMA interface for the Analog-to-Digital Converter (ADC). The DMA is designed by adding circuits to the interrupt serviced device interface. The interface is composed of two interface registers: the ADCSR register, which contains flag and error bits; and the combined ADBAR/ADMUX register, which holds the bus address and multiplexer bits.

4.5.1 Interface Description

Interface operation begins when the program loads the bus address register (ADBAR) with the address of the first memory location where data is to be stored. The interface starts an ADC conversion cycle. When the digital data is available from the ADC, the interface requests bus use by asserting an NPR request. When the device becomes bus master, it transfers the data to core memory. Completion of the bus transfer causes the multiplexer register (ADMUX) to be incremented, thereby selecting the next input channel. The multiplexer register is part of the bus address register; therefore, the next core memory location is also selected. At this point, a new conversion cycle begins. This process is repeated until each input channel is read and the digital data is stored in a core memory location. The interface then sets the ready flip-flop, which causes an interrupt.

4.5.2 Interface Implementation

The DMA interface is constructed by adding one set of bus drivers and the bus transfer control logic to the interrupt serviced interface; therefore, the functions assigned to the registers differ in this case, and implementation differs accordingly. The multiplexer register, expanded to 15 bits, also serves as a bus address register. Nine of these bits (15:07) are under program control and serve as a base address for a series of locations used as a data collection buffer by the interface. The remaining six bits (06:01) are implemented as a counter that steps through the 64 inputs and also addresses 64 successive word locations in the core memory. The six multiplexer bits are not accessible from the bus and cannot be read nor altered by the program. Whenever the high or low byte of the address register is loaded, the six multiplexer bits are cleared to zero; therefore, transfers always start on 64 word boundaries.

The interface uses an interrupt to signal completion of the series of transfers. The interrupt enable (INTR ENB) and READY bits of the ADCSR operate similar to the interrupt serviced interface.

Figure 4-9 is a timing diagram of the DMA interface. Figure 4-10 is a circuit schematic of the DMA interface, which is implemented entirely with standard Digital M-series modules. The basic control is provided by the M796 module. The multiplexer (ADMUX) counter is implemented by an M211 Binary Up/Down Counter Module.

Loading the ADBAR register (SELECT 2 · OUT HIGH or SELECT 2 · OUT LOW) also clears the multiplexer counter and the READY flip-flop, thereby initiating a conversion cycle by causing START CONVERSION H to become asserted.

When the conversion is complete, the CONVERSION COMPLETE H signal sets the REQUEST BUS flip-flop, which causes the M782 Interrupt Control to assert an NPR request. When bus control is granted, the M782 asserts BBSY on the Unibus and asserts the MASTER A L signal. The MASTER A L signal is tied to the M796 Unibus Master Control module in order to produce the START signal. Since the C1 control line is high and the C0 control

line is grounded, the M796 performs a DATO bus cycle. An ADRS TO BUS H is produced to gate the nine bits of the ADBAR register and the six bits of the ADMUX register to bus address lines A⟨15:01⟩. DATA TO BUS places the converted digital value on bus data lines D⟨09:00⟩. After a minimum delay of 150 ns, BUS MSYN L is asserted.

When the slave device responds with BUS SSYN L, both ADRS TO BUS and DATA TO BUS are negated and BUS MSYN L is dropped. The END CYCLE H pulse is used to clear the REQUEST BUS flip-flop, which in turn causes the M782 Interrupt Control to drop BUS BBSY.

END CYCLE L is used to trigger a one-shot to produce the COUNT DELAY H signal. This signal serves as the count input (COUNT IN) to the multiplexer counter (ADMUX). After 600 ns, the one-shot times out and its output returns to a low (0V) level. If the READY flip-flop has not been set by a count overflow from the ADMUX counter, START CONVERSION H is asserted to start the next conversion cycle. If, however, the ADMUX counter has overflowed and set the READY flip-flop, no ADC operation is started and an interrupt bus request is made.

A TIME-OUT flip-flop is provided on the M796 module. This flip-flop is set if the slave does not respond within 20 μs to the BUS MSYN L signal that is produced by the M796 module. If TIME OUT becomes set, the bus cycle is stopped, READY is set, and further conversions are inhibited. The TIME-OUT ERROR is indicated by a 1 in bit 15 of the ADCSR. TIME-OUT is cleared by loading bit 15 of the ADCSR with a 0.

The modules required to implement this interface fit into one BB11 System Unit. All interface modules, including the M782 Interrupt Control, M105 Address Selector, and a device cable connector, can be inserted into the logic slots of one system unit containing power and Unibus connectors. The BB11 System Unit is described in Paragraph 3.4.1

4.5.3 Programming the Interface

The following is an instruction sequence to initiate device operation:

```
MOV  #BUFADR, ADBAR      ;load address and start
MOV  #100, ADCSR         ;enable interrupt
```

where: BUFADR is the address of the first word of a buffer and is restricted to all 0s in bits 0 through 6.

The interrupt routine for this interface is equivalent to the data evaluation routine suggested in the interrupt serviced interface. The routine should begin with a CLR ADCSR instruction to disable further interrupts (unless serviced at a higher priority level) and should terminate with an RTI instruction.

The ADBAR register can be read as a source operand without spurious clocking of the device operation cycle, but the ADMUX counter is not accessible from the bus.

The interrupt enable flip-flop (bit 6 of the ADCSR) is entirely under program control but the TIME-OUT flip-flop is set by TIME-OUT ERROR conditions in the interface. The ready bit of the ADCSR (bit 7) is not under program control. It may be read by the program but cannot be altered except by initiating operation of the device.

4.5.4 Interface Operation Timing

Figure 4-9 illustrates the timing relationships among signals in the DMA interface. The curved lines indicate the changes in signal level that generate the indicated results.

4.5.5 Interface Options

As described above, operation of the DMA interface is restrictive, because it must always scan 64 channels. A simple method of reducing the number of channels scanned is to alter the set/reset inputs to the M211 Binary Counter module, thereby preloading it with a non-zero constant from which it can begin counting up.

An even more flexible arrangement could be designed by separating the ADBAR and ADMUX registers, thereby allowing independent bus addressing and multiplexer scanning.

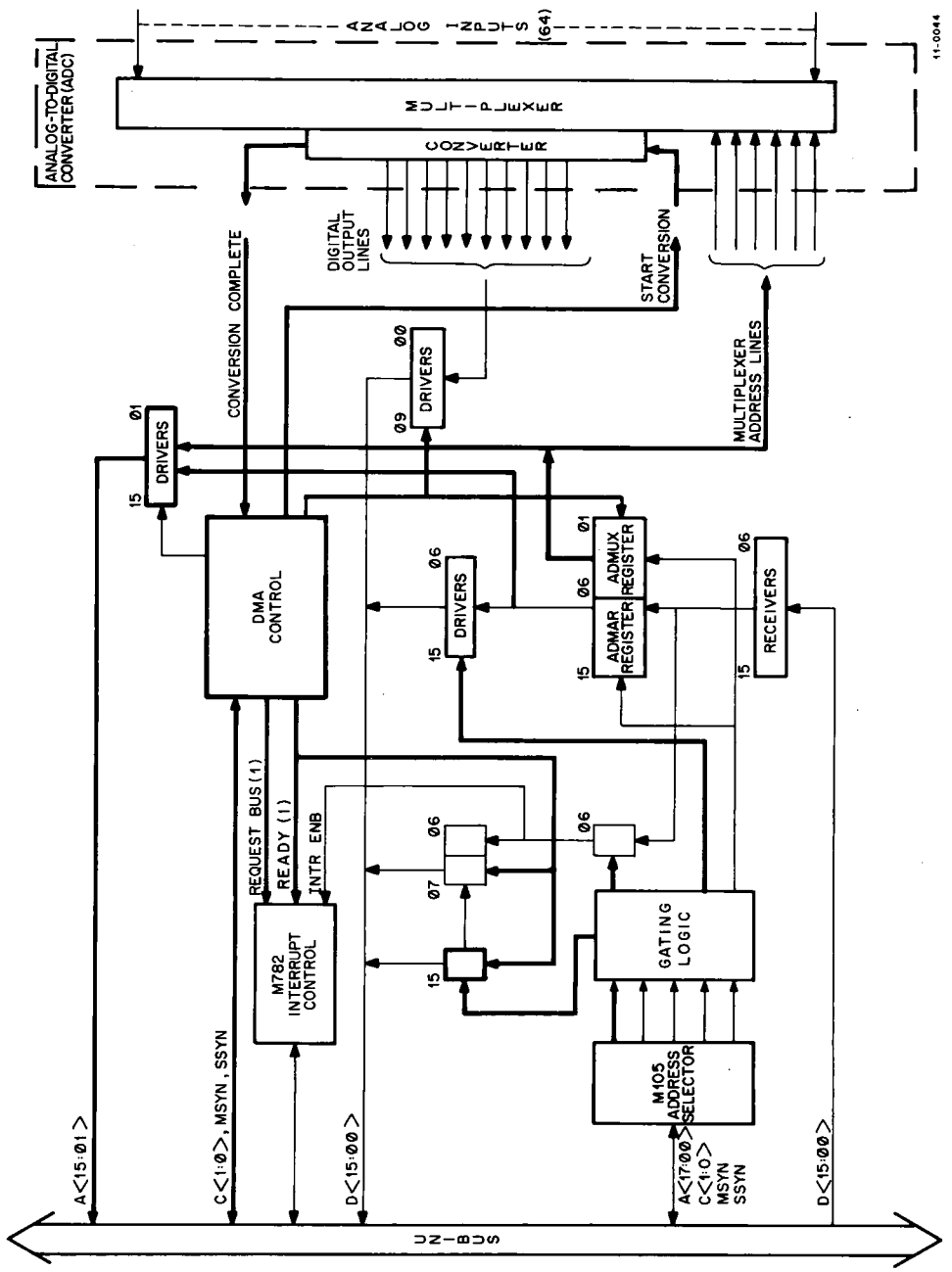


Figure 4-8 DMA Interface (block diagram)

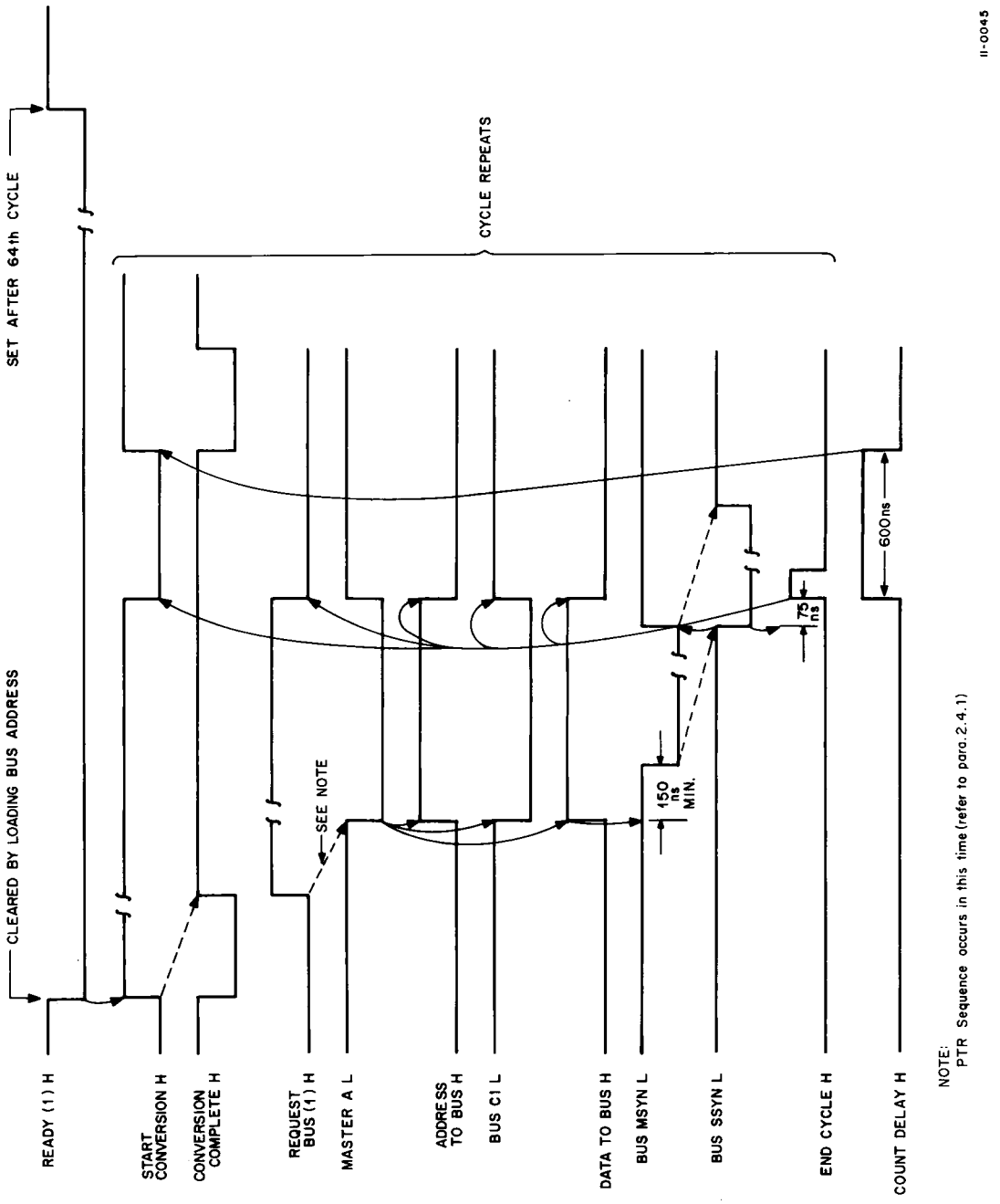


Figure 4-9 DMA Interface (timing diagram)

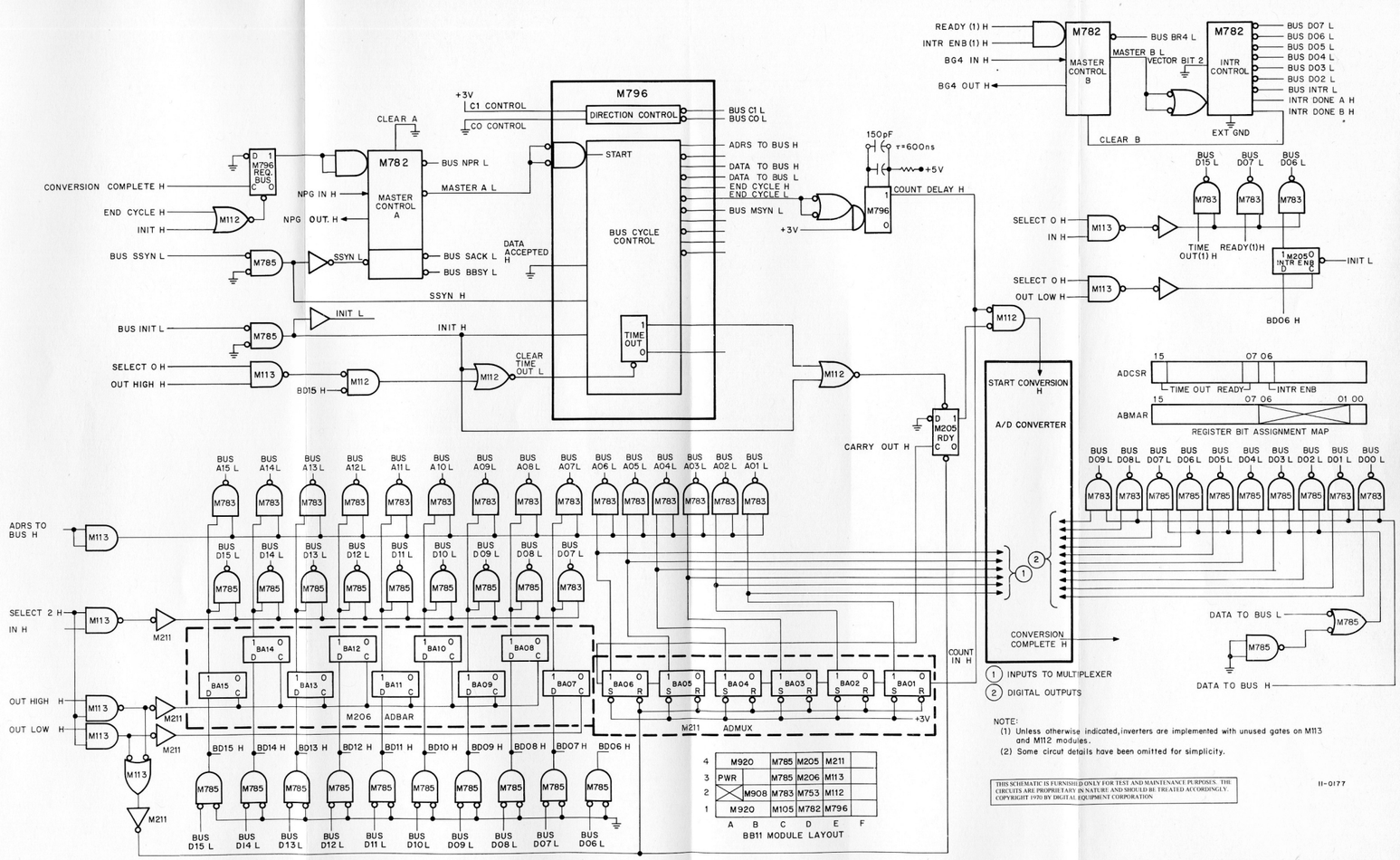


Figure 4-10 DMA Interface (schematic diagram)

4.6 OUTPUT INTERFACE WITH INTERRUPT CONTROL

Preceding examples have illustrated various types of interfaces for a peripheral device, which provides inputs to the Unibus data lines. This example, as well as the example in Paragraph 4.7, covers interface design for a device that accepts Unibus outputs. The device shown is meant to be typical of output devices which may be interfaced by designs similar to the following examples.

4.6.1 Device Description

A digital-to-analog converter (DAC) is a device that accepts Unibus outputs. The DAC converts a digital value to an equivalent analog voltage. The device is single-buffered and the analog output follows the digital input.

The interface provides 10 binary level inputs to the DAC. These inputs represent the digital value equivalent to the analog voltage desired as an output. The binary levels are 0V for binary 0 and +3V for binary 1.

The DAC provides an update request output signal for the interface. This signal requests a new digital input from the interface. At intervals determined by the DAC, a high level (+3V) pulse is provided as the update request signal. This level remains low (0V) between pulses.

4.6.2 Interface Description

The output interface with interrupt control provides a buffer register for outputs to the DAC and an interrupt control to service the DAC with an interrupt service routine. Figure 4-11 is a block diagram of the output interface.

The interface consists of two registers, an M105 Address Selector Module, an M782 Interrupt Control Module, bus receivers, and two sets of bus drivers. The two registers are the data buffer register (DADBR) and the control status register (DACSR). The request bit (bit 7) of the DACSR can be read by the bus but cannot be loaded directly from the bus. All other register bits are under direct bus control.

4.6.3 Interface Operation

When the Unibus addresses the data buffer register during a DATO transfer, the interface clocks the information from the bus data lines into the register, which then applies the information to the DAC as the 10 binary level inputs. At the same time data is clocked into the register, the REQUEST flip-flop (bit 7 of the DACSR) is cleared. After this transfer is complete, when the peripheral device requests another value, the REQUEST flip-flop is clocked high by an UPDATE REQUEST signal from the DAC. If the interrupt enable flip-flop (bit 6 of the DACSR) is set, the interface asserts the bus request line. On becoming bus master, the interface performs an interrupt operation to transfer program control to a service routine. This routine loads new data into the buffer register and then returns control to the interrupted program.

4.6.4 Circuit Implementation

Figure 4-12 shows the output interface with interrupt control, a bit assignment map for the registers, and a module layout of a BB11 System Unit. The interrupt control is similar to the control described in Paragraph 4.4. The buffer register is identical to the register described in Paragraph 4.2.

The only additional control logic required is the request flip-flop, which initiates a cycle of interface service and remains set until cleared by the loading of DADBR.

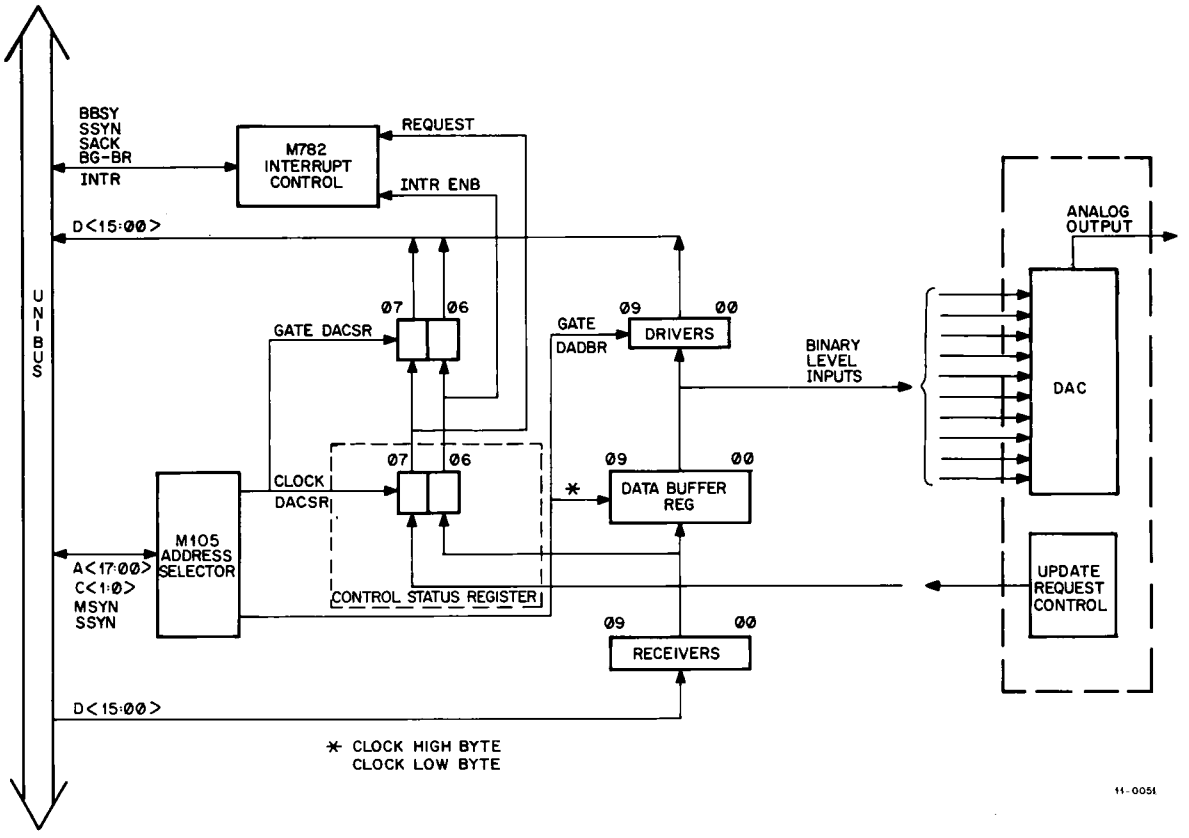


Figure 4-11 Output Interface with Interrupt Control (block diagram)

During normal operation, data is loaded into the buffer register and transferred to the peripheral device. When an UPDATE REQUEST from the DAC starts an interface cycle, the interrupt vector is transferred to the processor. The processor again initiates the data flow by transferring a new word of data into DADBR.

4.6.5 Interface Programming

The programs described in this paragraph cause the DAC to output a time-varying signal by loading the DADBR with an initial value and then changing that value by small increments until it reaches a final value determined by the program. The analog output is 100 cycles of a triangular waveform (actually, a stepped triangular waveform) with the slope of the ascending portion equal to half the slope of the decending portion. The period of the waveform is 150 times the period between update request pulses.

In the interface program, the DAC output is reset to a higher value by the ADD #10, DADBR instruction or reset to a lower value by the SUB #20, DADBR instruction. In either case, the value in the DADBR is read, modified by an arithmetic operation, and the new value is stored in the DADBR. All these operations are under processor control.

The ability of the Unibus to access device registers as though they were core memory locations allows the processor to directly perform tests and modifications on the device register. This program compares the value in the DADBR with the test values. The program uses a minimum of stored data because it is not necessary to use memory locations for counters or storage of temporary values.

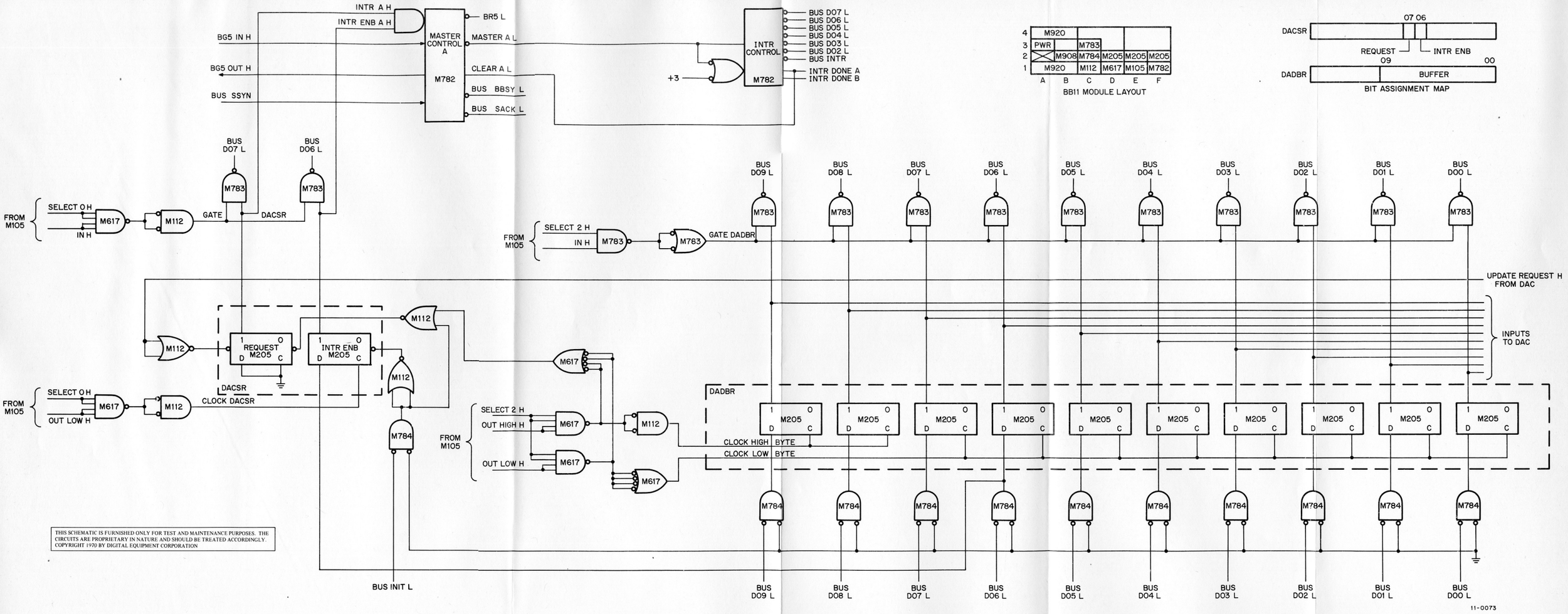


Figure 4-12 Output Interface with Interrupt Control (schematic diagram)

The processor initializes operation by executing the following sequence of instructions:

```
        CLR     DADBR           ;clear data buffer register
        CLR     DASW            ;reset up/down switch
        MOV     #144, DACNT     ;set cycle counter to 100
        MOV     #100, DACSR     ;set interrupt enable
```

The interrupt service routine includes the following instructions:

```
DAVEC:  DASERV                  ;pointer to service routine
        240                     ;processor priority = 5

DASERV: TST     DASW            ;switch set?
        BPL     UP              ;no, go up
        SUB     #20, DADBR      ;yes, go down
        BNE     CONT            ;output value equals 0?
        CLR     DASW            ;yes, reset switch
        DEC     DACNT           ;reduce count by one
        BNE     CONT            ;count equals 0?
        CLR     DACSR           ;yes, disable interrupt and exit
        RTI

UP:     ADD     #10, DADBR      ;output value goes up
        CMP     DADBR, #1000    ;1000 is top limit on value
        BNE     CONT            ;does value equal top limit?
        COM     DASW            ;yes, set switch

CONT:   RTI                     ;exit for intermediate values
```

4.7 DAC-DMA INTERFACE

A direct memory access (DMA) interface designed for a digital-to-analog converter (DAC) allows a specified number of words from memory to be transferred directly to the interface without processor intervention.

The previous interface example (paragraph 4.6) described a digital-to-analog converter interface that was serviced (controlled) by the vectored interrupt structure. In a real-time system where the time to service repetitive interrupts demands too much processor time, it may become necessary to expand the control section of the interface, so that the interface is less dependent on processor control, thereby reducing processor overhead.

This interface example uses the same DAC as the one discussed in the previous example. However, the interface to the Unibus differs. Added to the interface control section are direct bus access logic circuits, a word count register, and a bus address register. These additions allow a specified number of words from a particular group of memory addresses to be transferred directly to the interface, independent of processor control. This interface may be used, for example, to drive the X–Y deflection circuits of a CRT display scope in a refresh direct from memory mode.

4.7.1 Interface Description

A block diagram of the DAC-DMA interface is shown in Figure 4-13. The interface contains four registers: a DAC control and status register (DACSR) which contains control and status information; a DAC word count register (DAWC) which holds the 2's complement of the number of words to be transferred; a DAC bus address register (DABA) which indicates where the block of information is held in memory; and a DAC data buffer register (DADB) which buffers information during bus cycles and which can also be loaded under program control.

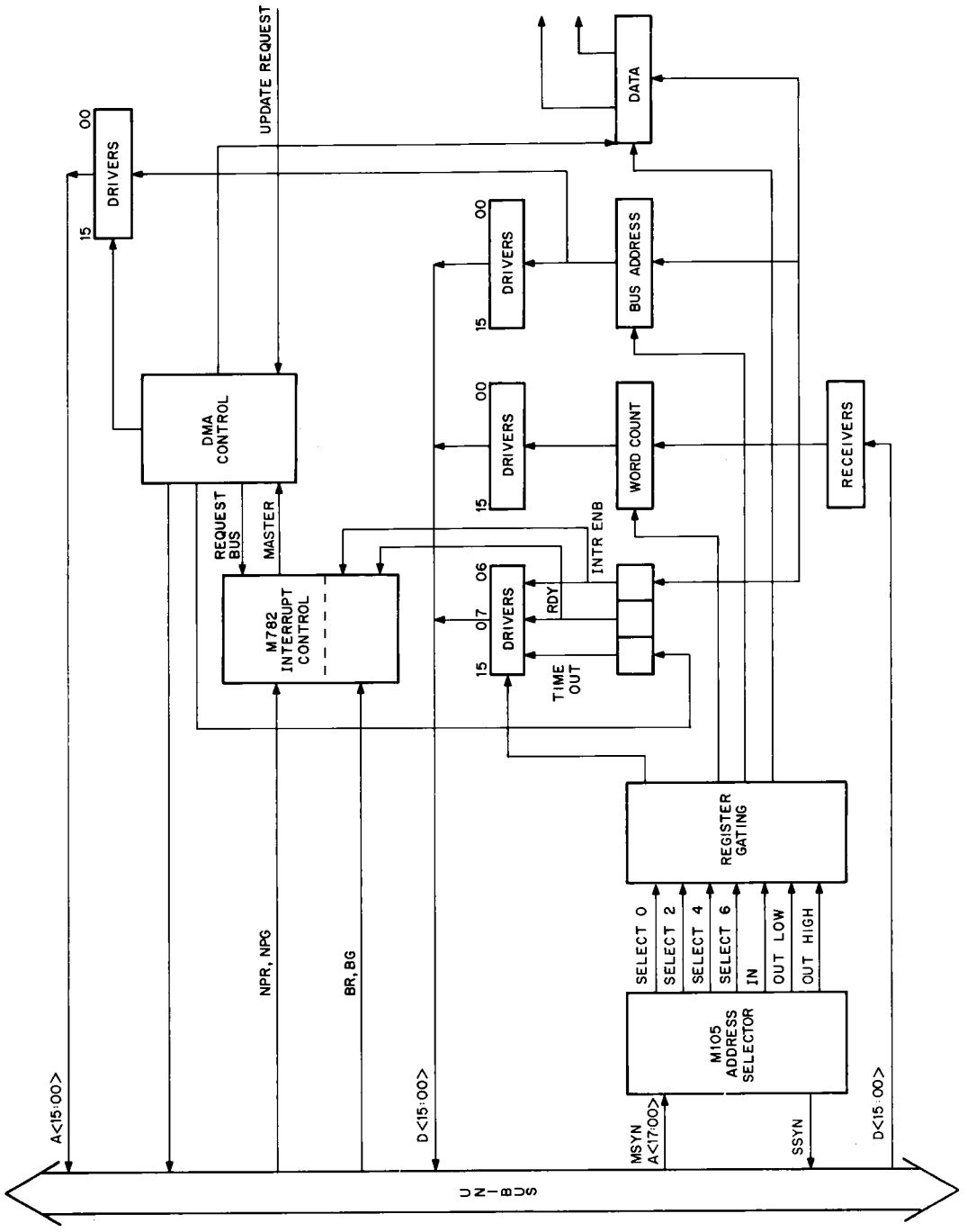


Figure 4-13 DAC-DMA Interface (block diagram)

A typical method of programming this interface is to first initialize the control by loading the DAWC and DABA registers. The next step is to set the GO bit in the DACSR. Words of data are then sequentially taken from memory and loaded into the DADB register at a rate set by the DAC or by an external clock. After each transfer (which is under the control of the interface rather than the processor), the DAWC and DABA registers are incremented. Data transfers continue until the DAWC register overflows (goes to all 0s). At this point, a READY bit in the DACSR is set. The READY bit can cause an interrupt to occur (provided INTR ENB is set), thereby notifying the processor that the block transfer is now complete, and another block transfer can be started.

4.7.2 Interface Operation

A circuit schematic of the DAC-DMA interface is shown in Figure 4-14 and a timing diagram is shown in Figure 4-15. Operation of the interface is started by loading bit 0 of the DACSR to produce the GO L signal which clears READY and allows the next UPDATE REQUEST H pulse (produced by the DAC or external clock) to set the REQUEST BUS flip-flop. When this flip-flop is set, the REQUEST BUS signal causes the M782 Master Control A to make a request for bus use on the BUS NPR L line. After the current bus master completes use of the bus, and the DAC-DMA interface receives an NPG signal, the interface gains control of the bus and asserts BUS BBSY L. Simultaneously, MASTER A L is asserted which produces the START H signal on the M796 Unibus Master Control module. Since both the C1 and C0 control lines are grounded, the M796 module performs a DATI bus cycle.

The ADRS TO BUS H signal is used to gate the contents of the DABA register onto the bus address lines. After a minimum delay of 150 ns, BUS MSYN L is asserted. Eventually, the addressed slave (usually memory) places the requested data on the bus data lines and responds with BUS SSYN L. The BUS SSYN L signal enters the M796 module as SSYN H and triggers the DATA WAIT L pulse. This pulse allows for settling and deskewing of data from the slave. The trailing edge of DATA WAIT L triggers the DATA STROBE one-shot to produce DATA STROBE L which clocks the data present on the bus D lines into the DADB register. The DATA STROBE H pulse is fed back into the M796 module as DATA ACCEPTED. This signal causes BUS MSYN L to be negated and also negates ADRS TO BUS a minimum of 75 ns later. At the conclusion of the bus operation, an END CYCLE pulse is produced. The END CYCLE H pulse clears the REQUEST BUS flip-flop and the M782 module therefore drops BUS BBSY. The END CYCLE L pulse increments the DAWC and DABA registers. If the DAWC register overflows, READY is set and subsequent UPDATE REQUEST signals are prevented from setting the REQUEST BUS flip-flop. If there is no overflow from the DAWC register, the interface waits for the next UPDATE REQUEST to start another cycle of operation.

If no slave responds to the address generated by the DAC-DMA interface, the M796 module sets a TIME-OUT flip-flop and does not perform a bus cycle. This error condition is available as bit 15 in the DACSR. The combination of READY and INTR ENB being set causes a vectored interrupt.

4.7.3 Interface Options

The interface shown in Figure 4-14 indicates that the DADB register is implemented as a full 16-bit register. This appears excessive for standard 10 or 12 bit DACs; normally, the extra bits would not be implemented. However, another method of using this interface is to use two 8-bit DACs to drive the X and Y deflection circuits of a CRT display. In this method, the two bytes of each word represent the coordinates of a point on the CRT face. Additional timing circuits are necessary in this instance to provide blanking and/or to intensify signals and to create an UPDATE REQUEST signal.

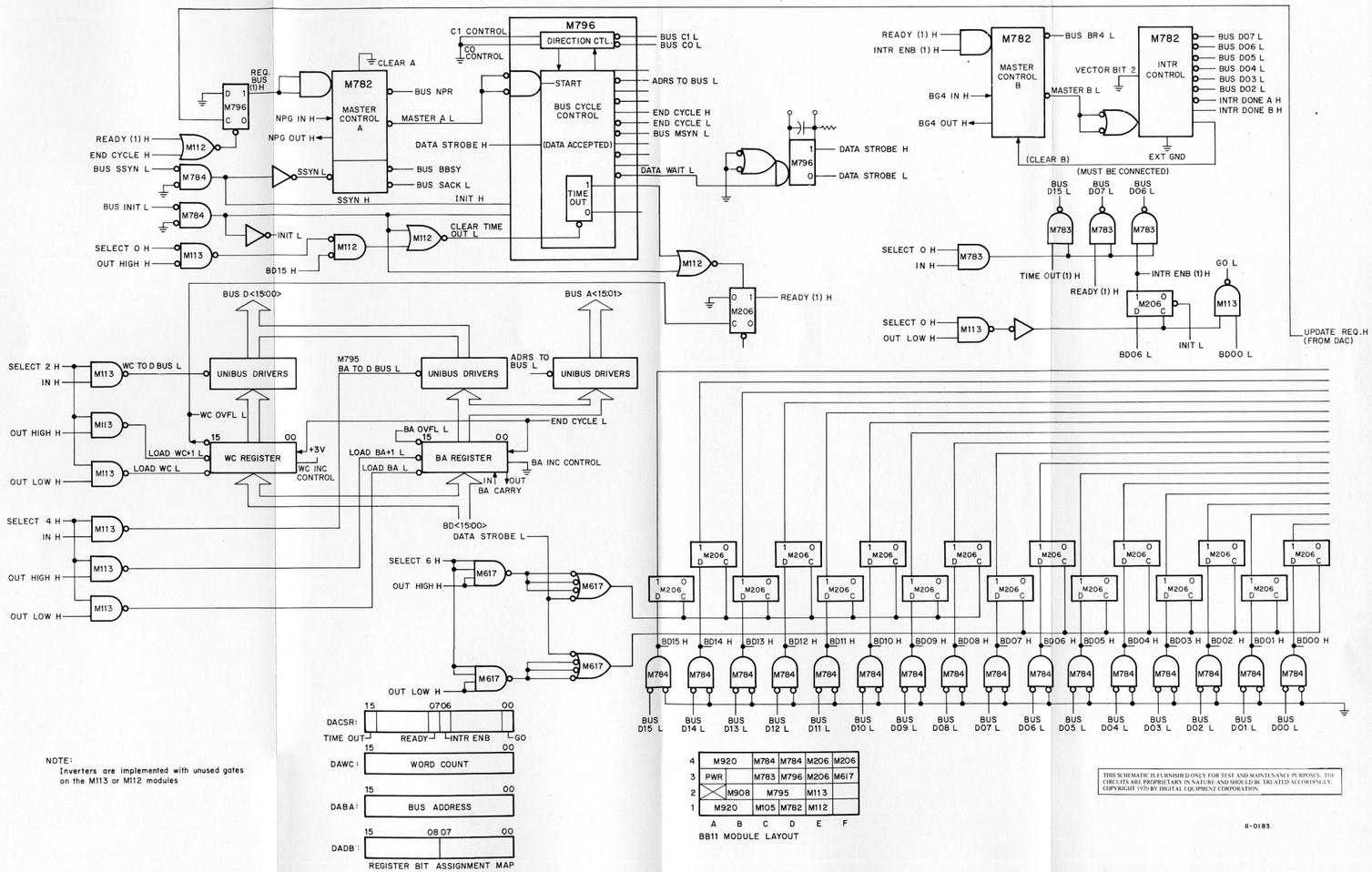


Figure 4-14 DAC-DMA Interface (schematic diagram)

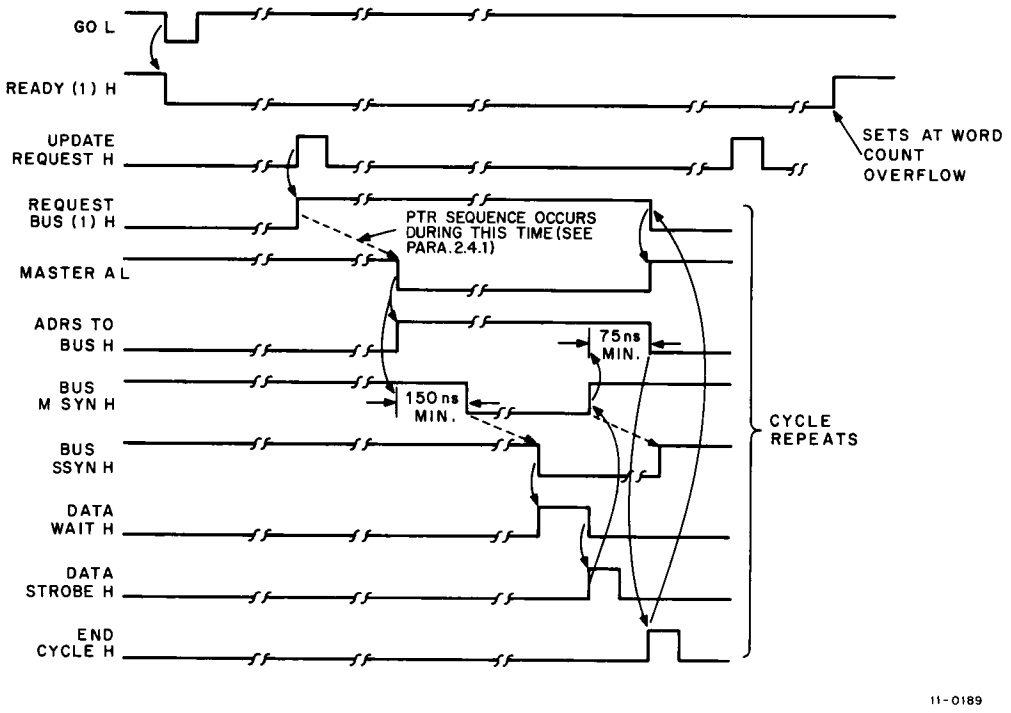


Figure 4-15 DAC-DMA Interface (timing diagram)

4.8 PDP-11 TO DATA CHANNEL INTERFACE

This paragraph presents a block diagram level description of an interface between the PDP-11 Unibus and a data channel of another computer. Asynchronous timing permits the PDP-11 to operate successfully with the timing of other systems through the implementation of a proper timing interface. Actual design depends on the characteristics of the other computer; therefore, this example is only a representative design.

4.8.1 Data Channel Description

Information on the data channel of the secondary computer includes an address, control and timing signals, and the data being transferred. The address is transmitted to a memory address (MA) register and is stored when the data channel begins operation. Data is supplied to a memory buffer (MB) register on one set of lines and is transmitted from the MB to the interface on a second set of lines. (These may be the same set of lines if the data channel uses bidirectional data lines.) Control and timing signals include: a break request (BRK REQ) signal; a data direction (DATA DIR) signal, which is high (+3V) for a transfer to the channel and low (0V) for a transfer from the channel; a break state (BRK STATE) signal, which is low when the secondary computer is in the break state; and three timing signals transmitted by the channel control to indicate break and address accepted (BRK ACPT), data accepted (DATA ACPT), and data available (DATA AVAIL).

4.8.2 Interface Description

The interface in Figure 4-16 operates as a cycle stealing device on a data channel of another computer and responds as a slave device on the PDP-11 Unibus. The primary purpose of this interface is to transfer data between two processors. Therefore, no provision is made to directly control operation of the secondary computer.

The interface structure includes the data paths between the two processors; the address paths from the Unibus to the secondary processor; a block (BLK) register (expandable to match the memory space of the secondary processor), which modifies the address transmitted on the data channel in order for it to access one of eight 4K word blocks; and the interface control logic, which determines the timing and gating of the data and control signal flow.

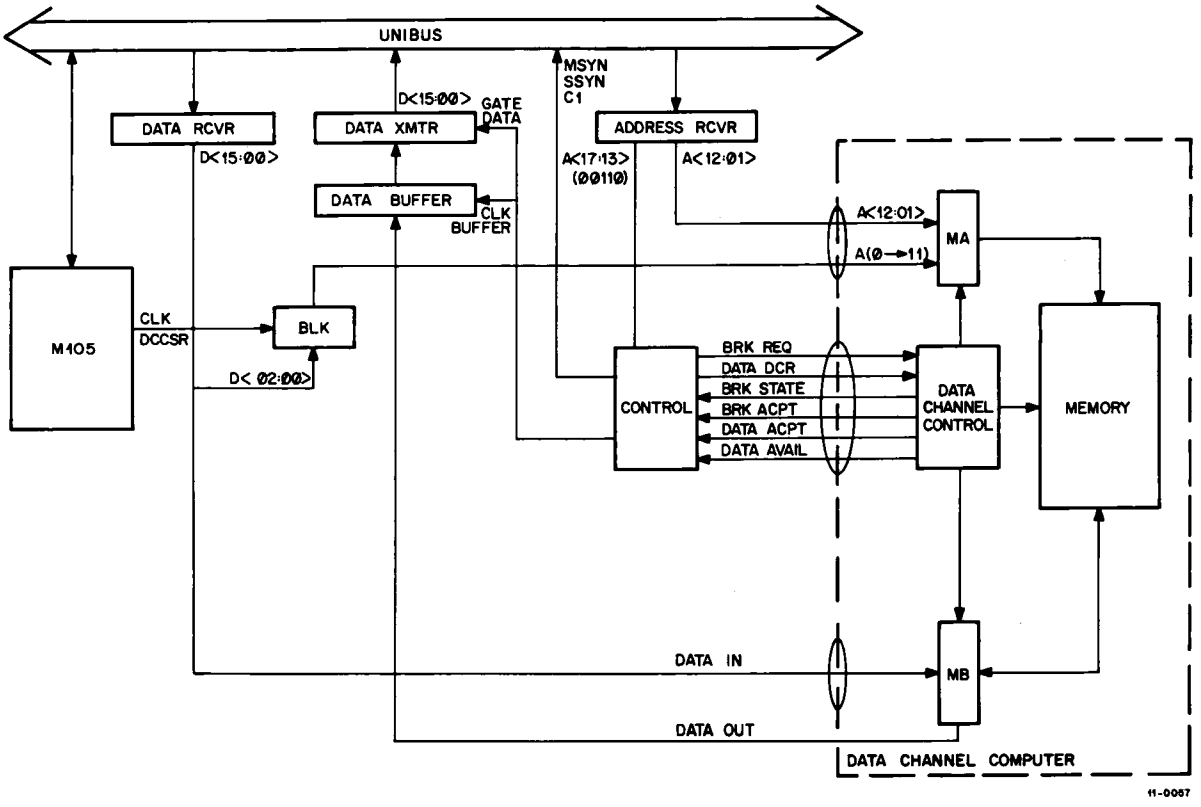


Figure 4-16 PDP-11-to-Data Channel Interface (block diagram)

When the PDP-11 addresses a bus location in the seventh block of 4K words (A⟨17:13⟩ = 00110), the interface generates a BRK REQ signal. The address used by the data channel is a 15-bit number assembled by transmitting the information on lines A⟨12:01⟩ as the 12 least significant bits, and transmitting the contents of the BLK register as the 3 most significant bits. The interface responds to 4K word (8K byte) addresses on the Unibus, but can access 32K words in the data channel. Only 4K words are accessible for any setting of the BLK register.

4.8.3 Interface Operation

When performing transfers with the data channel computer, the PDP-11 first loads the desired block number into the BLK register, which is addressed as a device register (A⟨17:13⟩ = 11111). Note that address lines A⟨12:00⟩ must be at the assigned bus address. The PDP-11 then addresses a location in the 4K block of Unibus addresses which have A⟨17:13⟩ = 00110. The corresponding location in the 4K block selected by the BLK register is selected by the data channel. A data transfer is accomplished by issuing a BRK REQ signal, an address, and setting the DATA DIR level; receiving a BRK ACPT pulse; gating data through the interface; and receiving either a DATA ACPT or a DATA AVAIL pulse to generate SSYN.

4.8.4 Interface Implementation

Figure 4-17 illustrates details of a possible control circuit for implementing the PDP-11 to data channel interface. The BRK REQ flip-flop is set when the bus address is in the specified 4K block and cleared by the BRK ACPT H pulse from the data channel. The SSYNF flip-flop is set by either a DATA ACPT H or a DATA AVAIL H pulse, depending on the direction of the data transfer. The SSYNF flip-flop is cleared when MSYN is cleared. The DATA AVAIL H pulse also clocks data into a buffer register, which is gated to the bus when the SSYNF flip-flop is set. This provides a data transfer to the bus. This implementation assumes that the data channel computer operates with logic levels similar to the PDP-11. Thus, a high (+3V) level is a logic 1 and a low (0V) level is a logic 0.

4.9 PDP-11 TO PDP-11 INTERFACE

The PDP-11 to PDP-11 interface permits devices on one PDP-11 Unibus to address locations on a second PDP-11 Unibus. Two possible applications of this interface are:

a. Connecting two PDP-11 processors so that one processor executes its own programs and also controls execution of programs by a second slave processor.

b. Monitoring operation of a Unibus system by a processor that is not in the same system.

4.9.1 Interface Description and Operation

The PDP-11 to PDP-11 interface (Figure 4-18) consists of one M782 Interrupt Control, one M105 Address Selector Module, several sets of bus receivers and bus drivers, and some control circuits. The M105 Module is connected to the primary Unibus and the M782 Module is connected to the secondary Unibus. The major flow of data within the interface is from one Unibus, through bus receivers and gated bus drivers, to the other Unibus. Data also flows from the primary Unibus to the control circuit.

When the primary Unibus requests interface operation, the interface requests control of the secondary Unibus, and the primary Unibus is connected to allow data transfers with the secondary Unibus. The interface is designed to recognize addresses in the seventh 8K (byte) field of primary Unibus addresses, and convert them to addresses in any selected 8K (byte) field of addresses on the secondary Unibus. This is equivalent to replacing one 8K field

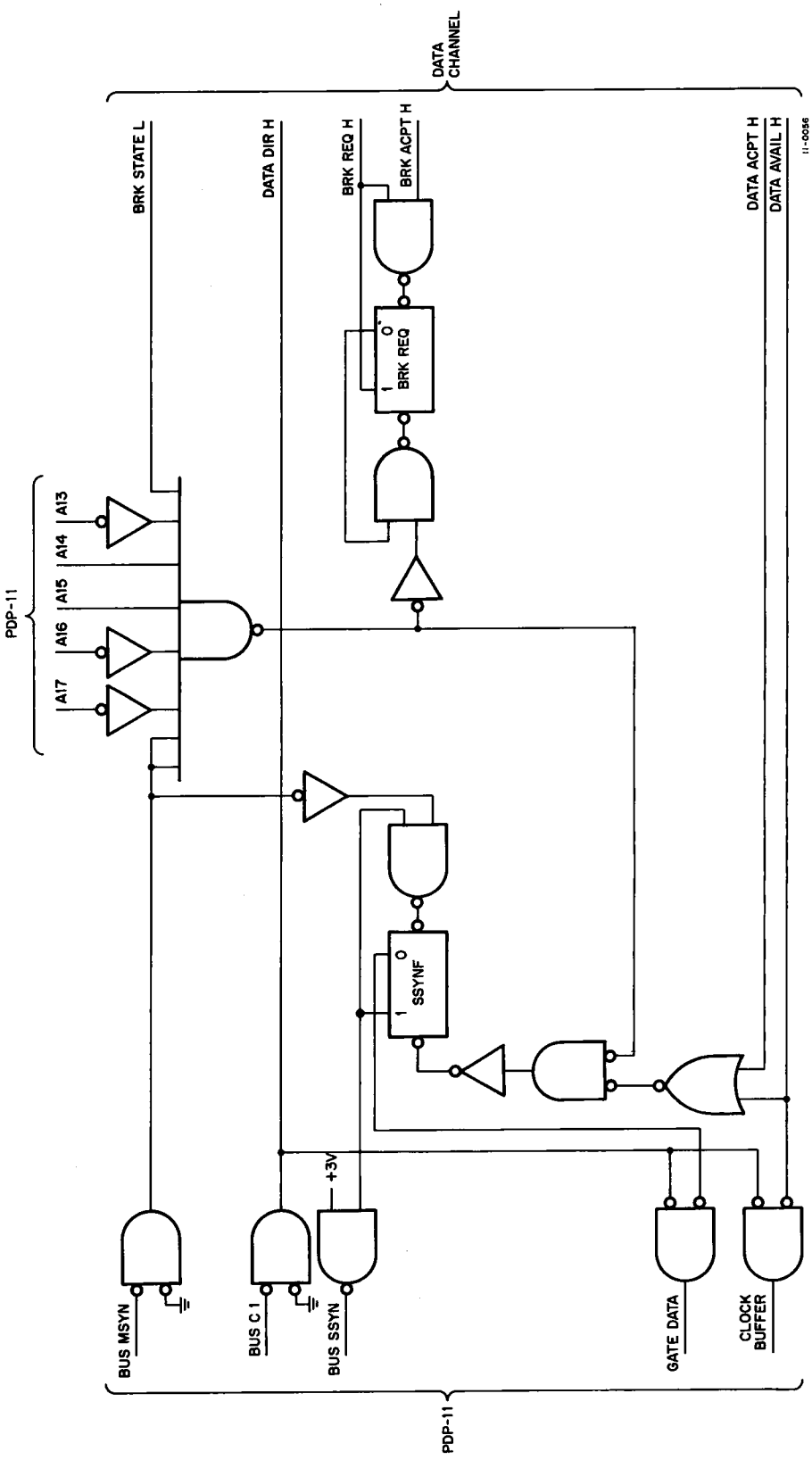


Figure 4-17 PDP-11-to-Data Channel Interface (control circuit schematic)

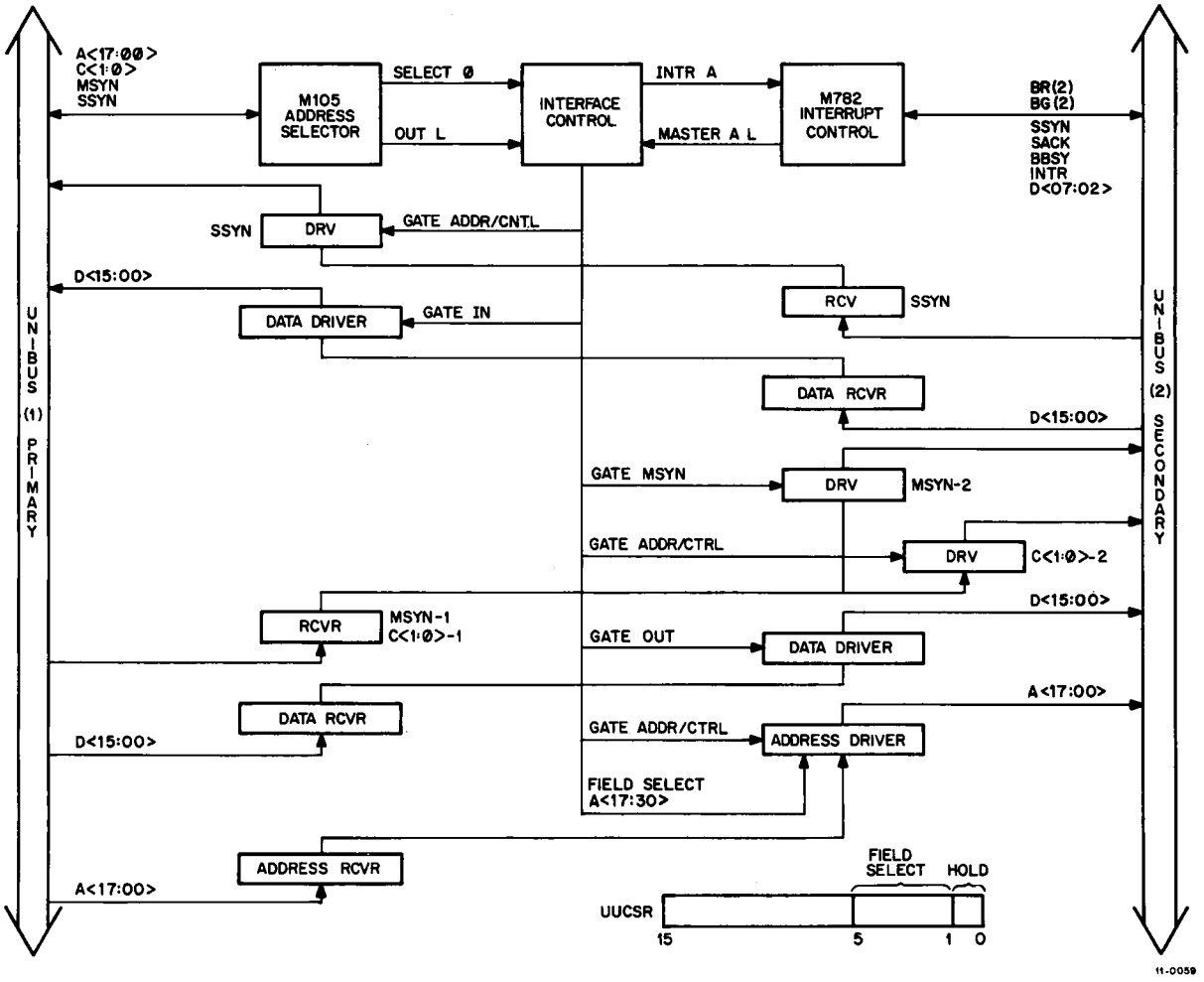


Figure 4-18 PDP-11-to-PDP-11 Interface (block diagram)

of bus addresses on the primary Unibus with an 8K field of addresses from the secondary Unibus. The field on the secondary Unibus is selected by loading a field select register in bits 5 through 1 of the Unibus-to-Unibus control status register (UUCSR) in the interface. The contents of this 5-bit register are used for the five most significant bits of the secondary bus address. This may be loaded by a MOV BLOCK, UUCSR instruction.

The interface can be operated in one of two modes. In the first mode, a single bus operation may access the secondary Unibus by addressing the seventh 4K field on the primary Unibus. The interface recognizes the address, requests control of the secondary bus, and interconnects the two sets of bus lines when it receives control. The interface releases control of the secondary Unibus when the transfer is complete.

In the second mode of operation, if the hold bit is set in the UUCSR, the interface requests control of the secondary bus and maintains bus mastership until the hold bit is cleared. This mode of operation permits the primary bus to conduct a series of data transfers with the secondary bus at the full bus transfer speed. The only additional time required for a transfer is the signal transmission time within the interface. The time delays caused by waiting for the secondary bus to grant bus mastership to the interface are eliminated; therefore, a disk on the primary bus can transfer data to locations on the secondary bus with the lowest possible latency delays.

4.9.2 Interface Implementation

In Figure 4-18, Unibus 1 (the primary Unibus) controls the interface and conducts data transfers through the interface to Unibus 2 (the secondary Unibus). The interface control circuit is shown in greater detail in Figure 4-19. This figure illustrates the UUCSR device register, an address recognition circuit, a MSYN delay circuit, and several gating circuits.

When the address asserted on the primary Unibus is in the seventh 8K field (A⟨17:13⟩ = 00110), the STEAL signal is asserted in the interface. The STEAL L signal causes an NPR request on the secondary Unibus. (If the hold flip-flop is set, the interface is already bus master and no further bus requests are initiated.) When the interface becomes a bus master, the MASTER A L signal is asserted. The combination of STEAL L and MASTER A L produces gating signals to interconnect the two sets of bus lines as shown in Table 4-1. The MSYN delay circuit regenerates the 150-ns deskewing period between the assertion of the A and C lines and the assertion of MSYN on the secondary Unibus. This prevents delays in signal transmission within the interface from affecting the timing on either Unibus.

Table 4-1
Bus Line Gating

Signal	From	To	Gated by	Remarks
A⟨17:13⟩	Block Register	Unibus 2	GATE ADDR/CTRL	
A⟨12:00⟩	Unibus 1	Unibus 2	GATE ADDR/CTRL	
C⟨1:0⟩	Unibus 1	Unibus 2	GATE ADDR/CTRL	
D⟨15:00⟩	Unibus 2	Unibus 1	GATE IN	C1 = 0 (DATI or DATIP)
D⟨15:00⟩	Unibus 1	Unibus 2	GATE OUT	C1 = 1 (DATO or DATOB)
MSYN	Unibus 1	Unibus 2	GATE MSYN	
SSYN	Unibus 2	Unibus 1	GATE ADDR/CTRL	

4.9.3 Interface Programming

To type a character on the secondary Unibus system Teletype, the processor on the primary Unibus executes the following sequence of instructions:

```
MOV     #76, UUCSR
MOV     CHAR, XPB
```

where: UUCSR is the control register in the interface
CHAR is the location containing the character
XPB has the value 157566

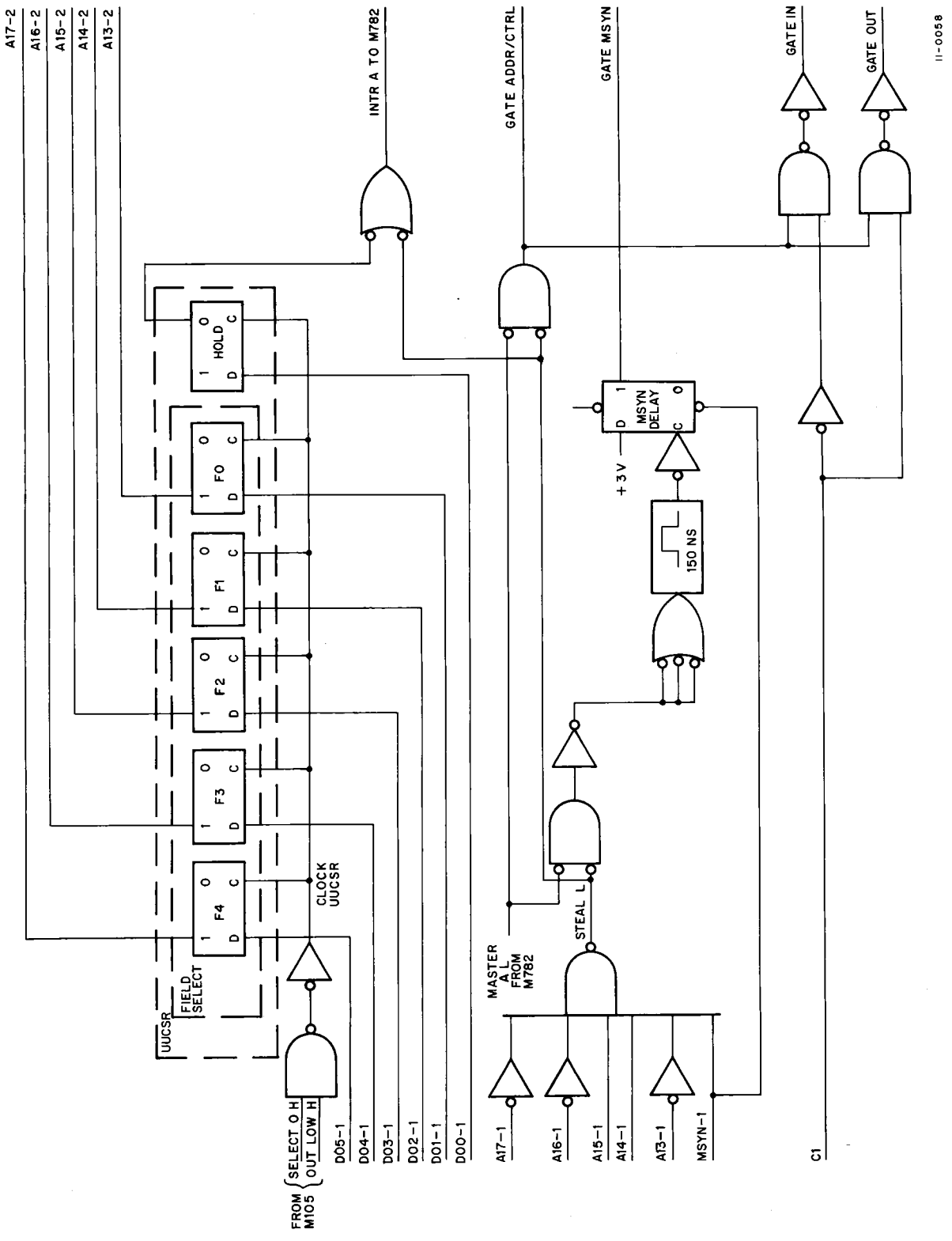


Figure 4-19 PDP-11-to-PDP-11 Interface Control (circuit schematic)

The address transmitted on the secondary Unibus is the 13 least significant bits of XPB (17566) with the contents of the field select register appended as the 5 most significant bits. The address is therefore 777566, which is the address of the system Teletype TPB.

A block of data is transferred to locations on the secondary Unibus by setting the hold bit when the field register is set:

```
MOV     FIELD, UUCSR     where: FIELD is the number of the 4K field of word addresses to be accessed on the secondary Unibus
                                (with a 1 appended to set the hold bit)
```

Data transfers then access locations in that address field when the address on the primary Unibus has A⟨17:13⟩ equal to 00110 and A⟨12:00⟩ equal to the desired value of A⟨12:00⟩ on the secondary Unibus. If FIELD has the value 11_8, the address 140020_8 is translated to the address 100020_8.

4.10 MEMORY INCREMENT INTERFACE

An Increment Memory interface is used to increment a device-selected memory location without requiring processor intervention. The interface operates by performing a DATIP, DATO sequence with the selected memory location. This sequence is the same as processor operation with a destination operand. Data read from the memory location by the DATIP transfer is loaded into a counter register. A count cycle is initiated, and the count is allowed to ripple through the register. The new value is written back into the memory during the DATO cycle.

A more flexible implementation uses an adder instead of a counter. The contents of the memory location are brought into a latch, which forms one input to the adder. The other input can be under device or program control, or a simple increment operation can be executed by providing a carry into the least significant bit. The adder permits arbitrary values to be added to the selected location. This expanded interface is not described in subsequent discussions.

This interface differs from previous interface examples because the device output is used as a bus address rather than as data. The operations performed on the data in these locations effectively generate a time-interval histogram directly in core memory.

4.10.1 Interface Description and Operation

In Figure 4-20, the analog-to-digital converter (ADC), which is an external device that supplies the digital value used as an address, is not part of the interface. One device register is used in the interface. The device register contains the enable bit which controls access of the interface to the Unibus, and contains the five field select bits used to select a 4K word block of bus addresses that the interface uses for conducting transfers. The counter and transmitters are under interface control. The interface provides all timing and control signals for one cycle of operation and each successive cycle of operation is initiated by a signal from the device. In this example, the device is an ADC similar to the ADC described in Paragraph 4.4. The rate at which successive conversion cycles are performed is not defined in this example. However, the rate can be controlled by an external clock, so that the clock frequency determines the sampling rate.

A timing diagram of the increment memory interface is shown in Figure 4-21 and a circuit schematic is shown in Figure 4-22. The signal that starts a cycle of interface operation is CONVERSION COMPLETE H. This signal is generated by the ADC after it has completed conversion of the analog input to a digital value, and that value is

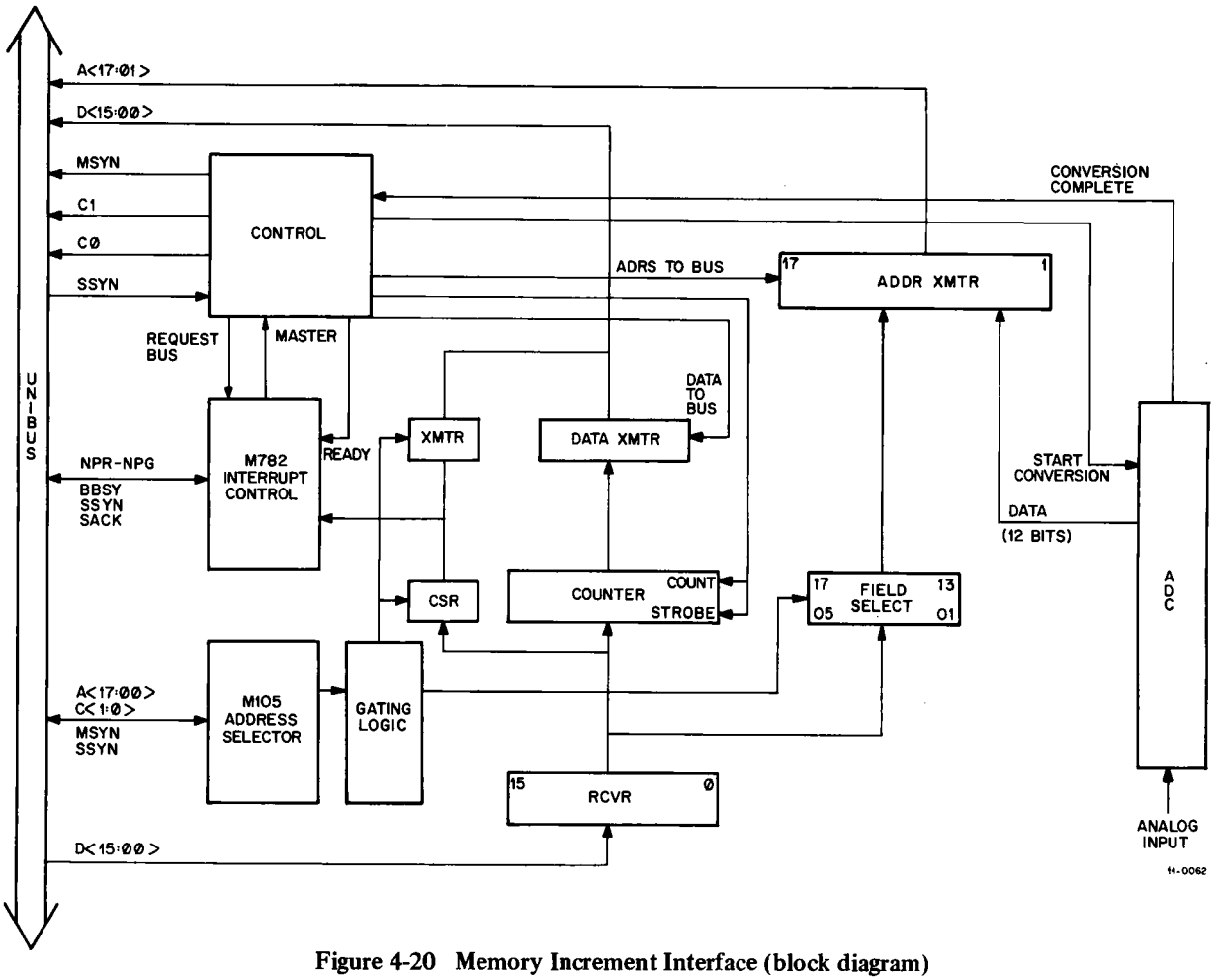


Figure 4-20 Memory Increment Interface (block diagram)

available as a data output from the ADC. The CONVERSION COMPLETE signal sets the REQUEST BUS flip-flop which causes the M782 Master Control A module to gain control of the Unibus by means of an NPR request. The MASTER A L signal triggers START on the M796 module and, since CYCLE CONTROL is clear, a DATIP bus cycle is performed. The ADRS TO BUS H signal is used to gate an address on to the address bus. This address is formed by data from the ADC (least significant portion of the address) and the field select bits (most significant portion of the address). After a delay, BUS MSYN L is asserted.

When the slave (usually memory) responds with data and a BUS SSYN L signal, DATA WAIT L is triggered to allow for deskewing of the data. The trailing edge of DATA WAIT triggers the DATA STROBE one-shot. The DATA STROBE signal loads the data present on BUS D⟨15:00⟩ into the interface counter register. When DATA STROBE times out, DATA ACCEPTED is asserted, negating BUS MSYN, and firing the COUNT PULSE and COUNT DELAY one-shots. COUNT PULSE is tied to the counter register and is used to increment the contents of the register. COUNT DELAY is set for the worst case ripple time of the counter.

When the COUNT DELAY is complete and BUS SSYN L is negated by the previous DATIP bus cycle, START is again asserted. Since CYCLE CONTROL is now set (set by END CYCLE of the previous DATIP bus cycle), a DATO bus cycle is performed. The DATA TO BUS H signal is asserted and gates the contents of the incremented counter to the bus data lines.

When the DATO cycle is complete, END CYCLE and CYCLE CONTROL set are ANDed to clear the REQUEST BUS flip-flop, which causes the M782 module to drop BUS BBSY L, thereby releasing control of the bus.

4.10.2 Optional Arrangements

The timing chain of DATA WAIT, DATA STROBE, COUNT PULSE, and COUNT DELAY is designed for the case where count time is significant and the counter is loaded and incremented by a level rather than a transition.

Other logic may be used dependent upon the type and speed of the counter employed in the interface.

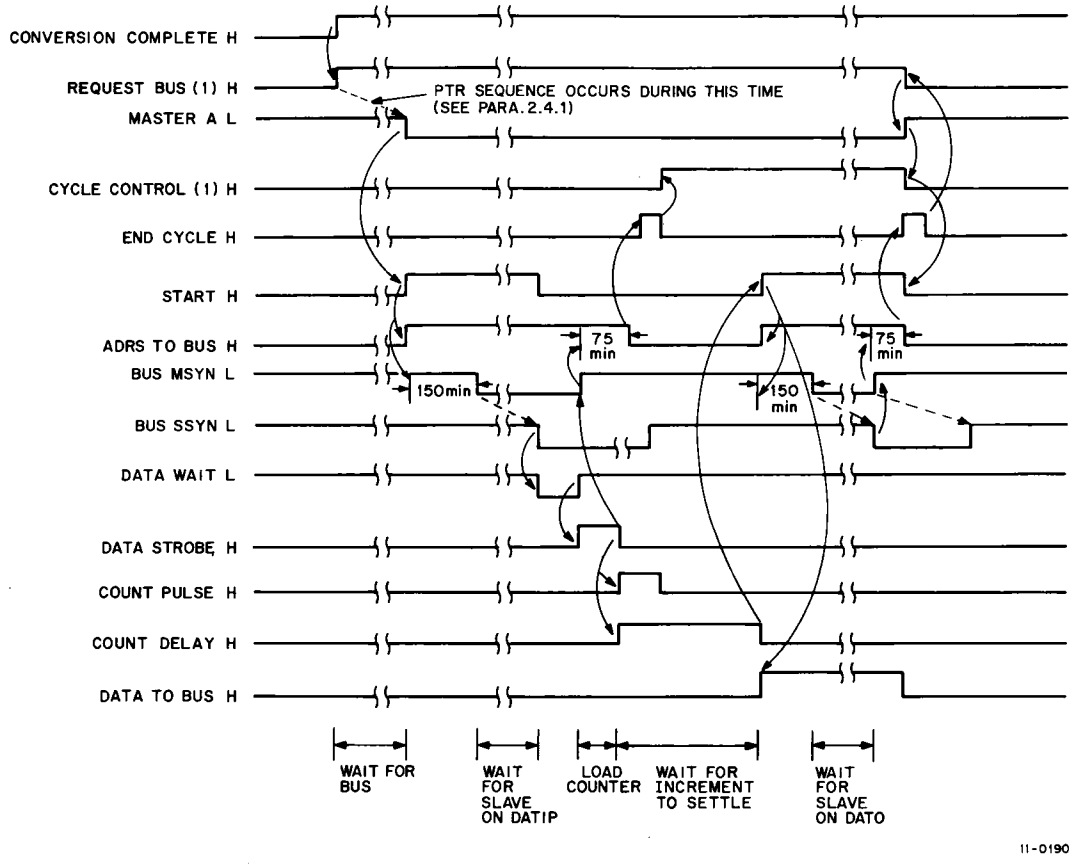


Figure 4-21 Memory Increment Interface (timing diagram)

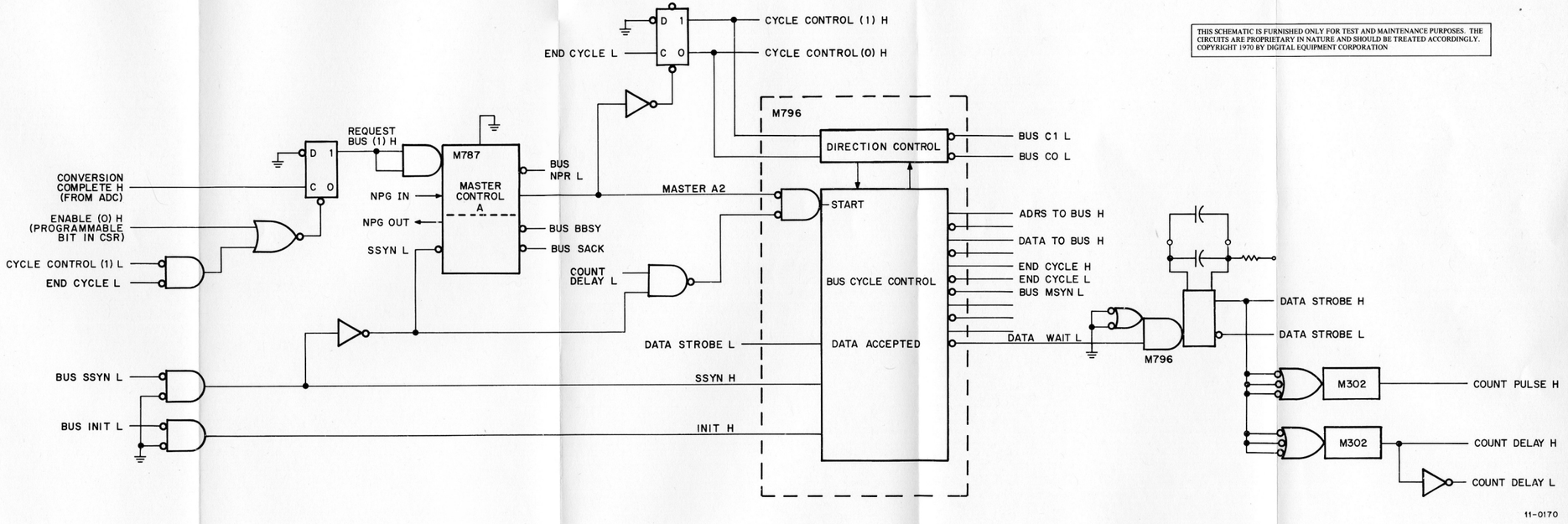


Figure 4-22 Memory Increment Interface (control circuit schematic)

APPENDIX A
PROCESSOR BLOCK DIAGRAM

A.1 INTRODUCTION

This description of the *KA11 (or KC11) Processor* is intended to give the reader sufficient knowledge of the processor, to understand the processor's role in the control of bus activity, and the use of bus operations involved in normal processing. It is beyond the scope of this Appendix to provide a detailed description of the processor. A detailed description of the processor, including theory of operation and logic design, is provided in the *KA11 Processor Manual,* DEC-11-HR2A-D. This Appendix describes the block diagram, and then briefly discusses major states and bus cycles.

A.2 BLOCK DIAGRAM

Figure A-1 is a simplified block diagram of the processor and is divided into four major functional areas: Unibus interface, data paths, general-purpose registers, and timing state and control logic. Each of these areas is discussed separately in subsequent paragraphs to bring out the logical structure of the processor.

A.2.1 Unibus Interface

The processor is connected to the bus data and address lines through input buffering and output driving gates. The bus control and control transfer lines are connected to the bus through the processor timing state and control logic. Most of the data originates, and ultimately is stored at or received by, locations external to the processor. Therefore, during normal processing the majority of information transfers take place through the bus interface.

A.2.2 Data Paths

The data paths perform all modification and routing of data within the processor. In effect, this is where normal processing and computation occurs. The data paths consist of A and B input gating, an adder, and output gating.

A.2.3 General-Purpose Registers

In general, data storage is external to the processor, and these storage locations are accessed through the Unibus. The only internal storage locations available for general data, are the flip-flop storage elements (general-purpose registers). These registers are used as index registers for address calculations, or as storage for frequently used data. Six of these registers are used by hardware functions, one is used as a program counter, and one is used as a stack pointer.

All information transfers are through the data paths. The bus interface and general-purpose registers interconnect only through the data paths, so that frequent transfers of data set up a flow of information in a figure-eight pattern (see Figure A-2). This flow is from the Unibus to the bottom of the data paths, from the top of the data paths to the general-purpose registers, from the general-purpose registers to the bottom of the data paths, and from the top of the data paths to the Unibus.

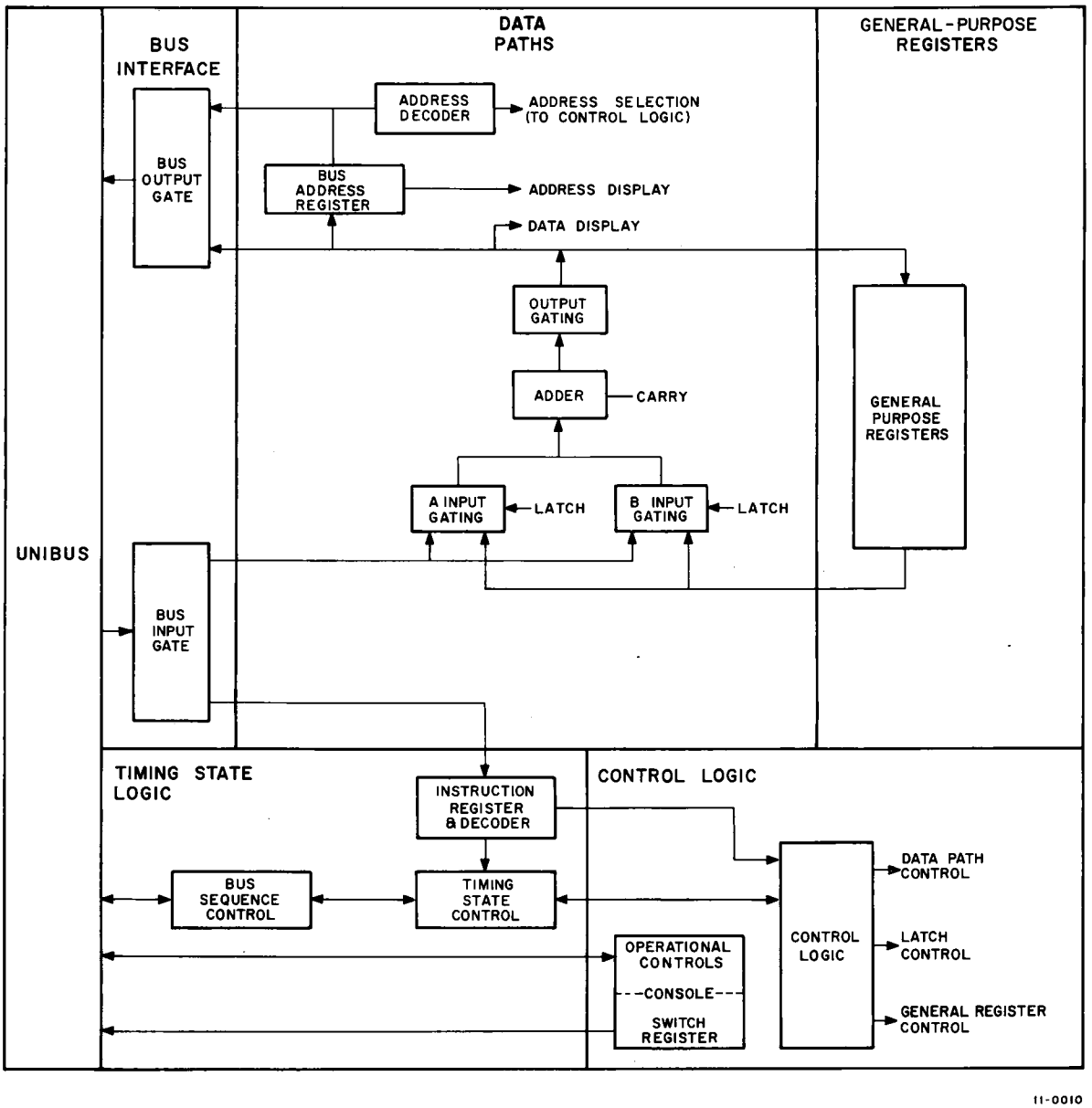


Figure A-1 PDP-11 Processor Block Diagram

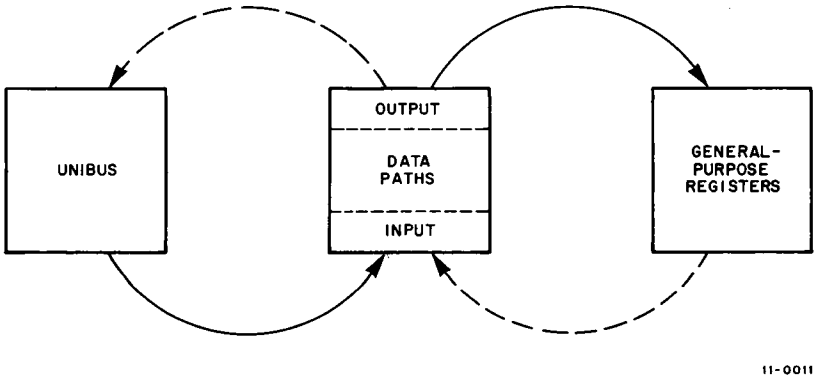


Figure A-2 Data Flow

A.3 MAJOR STATES

The *KA11 (or KC11) Processor* has five major states: fetch, source, destination, execute, and service. The first four states are used during normal processor operation, while the last state (service) is used during special operations, such as traps and interrupts. A brief description of major states and related functions is presented in Table A-1.

Table A-1
Processor Major States

Major State	Operand	Function
Fetch	All instructions	Locates and decodes instruction.
Source	Double-operand group only	Decodes source field and uses data paths to transfer addressed data to proper location.
Destination	Double-operand group	Decodes destination field and uses data paths to transfer addressed data to proper locations. Must follow source major state in this instance.
	Any other group	Decodes destination field of instruction and transfers data to appropriate location. Other groups with destinations are: all single-operand, rotate/shift, and subroutine call. Also, jump and swap byte instructions of the operate group.
Execute	Most instructions	Performs operations specified by the instruction (examples: arithmetic operations, tests, etc.)
Service	Any instruction	Performs special operation such as interrupts and traps.

The fetch major state locates and decodes an instruction. When this major state is completed, the processor enters another major state, depending on the type of instruction decoded. It is possible to go from fetch to any other state, including back to fetch. Every instruction must first enter the fetch major state.

The source major state decodes the source field of a double-operand group instruction and transfers data to appropriate locations. The source major state is entered only if the instruction belongs to the double-operand group.

The destination major state decodes the destination field of the appropriate instruction. Destination fields are present in all double-operand, single-operand, rotate/shift, and subroutine call instructions.

The execute major state uses the information from previous major states to perform the specified operation. During this state, arithmetic operations, logic functions, and tests are performed.

The service major state is used to execute special operations, such as interrupts, traps, etc.

Some typical major state sequences are given in Table A-2.

Table A-2
Typical Major State Sequences

Instruction	Major States				
	Fetch	Source	Dest.	Execute	Service
DOUBLE-OPERAND GROUP (single-word format)					
1. Both source and destination address modes 1,2, or 3	X	X	X	X	
2. Both source and destination address modes are 0	X			X	
3. Only source address mode is 0	X		X	X	
4. Only destination address mode is 0	X	X		X	
DOUBLE-OPERAND GROUP (2- or 3-word formats)	X	X	X	X	
SINGLE-OPERAND GROUP	X		X	X	
TRAP GROUP	X				X
BRANCH GROUP (condition met)	X			X	
BRANCH GROUP (condition not met)	X				

A.4 BUS CYCLES

Whenever data is transferred between devices, either a DATI (data transfer in) or DATO (data transfer out) bus cycle is used. The direction of data transfers is relative to the master (controlling) device.

In addition to the basic DATI and DATO bus cycles, there is also a modified DATI, known as DATIP (data transfer in, pause) and a modified DATO, known as DATOB (data transfer out, byte).

During the fetch major state, the processor always performs one DATI bus operation. For the source and destination major states, when applicable, the processor may do from one to three DATI operations in each major state, depending on the address mode and the number of levels of indirect addressing. If no indirect addressing is used, the major state is skipped for that word of data.

The execute and service major states vary for different instructions. The states are shown, along with the related instructions, in Appendix B.

APPENDIX B
INSTRUCTION SET

B.1 INTRODUCTION

This Appendix provides a description of the major states and bus cycles used by each instruction in the *PDP-11 Instruction Set.* The discussion is divided into two parts: a brief description of the six groups of instructions, and a discussion of bus cycles used by each instruction.

A detailed description of processor major states, bus cycles, and instruction decoding is contained in the *KA11* or *KC11 Processor Manual.* A detailed description of the instruction set is given in the *PDP-11 Handbook.*

B.2 INSTRUCTION GROUPS

The *PDP-11 Instruction Set* is divided into six major groups. A brief description of each group is given below. Specific instructions within each group are covered in Paragraph B.3.

a. *Double-Operand Group* – These instructions permit certain functions to be performed on both the source and destination data. Typical functions include: move, compare, add, and subtract. Instructions in this group can specify either byte or word operands except add and subtract.

b. *Single-Operand Group* – These single-operand instructions permit certain functions to be performed on the destination data. Typical functions include: clear, complement, increment, decrement, etc. The operand may be either a byte or a full word.

c. *Conditional Branches* – These test instructions check the results of an operation and cause the program to branch according to test results. Typical tests include: equal to zero, overflow, carry, etc.

d. *Operate Group* – These direct instructions, such as HALT and WAIT, also contain trap instructions, and the Return From Interrupt (RTI) instruction.

e. *Subroutine Group* – This group contains the two instructions for using subroutines: Jump To Subroutine (JSR) and Return From Subroutine (RTS).

f. *Condition Codes Group* – These instructions set condition code bits in the status hardware register to define the processor status (PS) word. The PS indicates whether or not certain conditions (such as zero, carry, etc.) are present.

B.3 MAJOR STATES

A general description of processor major states is given in Appendix A. The prime purpose of Appendix B is to cover specific instructions and list the bus cycles that occur as the instruction enters each major state.

B.3.1 Fetch

All instructions must enter the fetch major state. A DATI is performed to bring the instruction into the processor. The instruction is decoded, and the processor enters any one of the other four major states, depending on the instruction. If it is a double-operand instruction, it can enter the source major state. If it is a single-operand, rotate/shift, Jump (JMP), Jump To Subroutine (JSR), or Branch (BR), it can enter the destination major state. If it is a branch with the condition met, internal address, RTI, or RTS, it enters the execute major state. If it is an interrupt or trap, it enters the service major state.

B.3.2 Source and Destination

Both the source and destination major states are used for address calculations. When address mode 0 is used in either of these states, the state is skipped because address calculation is not necessary. Bus cycles for the remaining 7 address modes are listed in Table B-1.

Table B-1
Address Modes
(Source or Destination)

Address Mode	Assembler Mnemonic	Source Operand	Destination Operand		
			Normal*	CMP, BIT, TST	JMP, JSR
1 2 4	@R or (R) (R)+ -(R)	DATI	DATIP – DATO(B)	DATI	NONE
3 5 6	@(R)+ @-(R) X(R)	 DATI DATI	 DATI DATIP – DATO(B)	 DATI DATI	 DATI NONE
7	@X(R) or @(R)	DATI DATI DATI	DATI DATI DATIP – DATO(B)	DATI DATI DATI	DATI DATI NONE

*Includes MOV, CLR, and all other instructions not mentioned in the last two columns of this table. The final cycle may be either a DATO or DATOB depending upon word or byte operation.

X = index word

B.3.3 Execute

Since most of the instruction processing takes place during execute, it is the most complex major state. Six tables (B-2 through B-7) provide coverage of bus cycle information for each instruction. Each table represents a specific group of instructions, and lists all instructions within that group. The tables list the major states each instruction enters, and the bus cycles that occur during execute. These tables are:

Table B-2	Double-Operand Group
Table B-3	Single-Operand Group
Table B-4	Operate Group
Table B-5	Subroutine Group
Table B-6	Condition Code Operators
Table B-7	Conditional Branches

Table B-2
Double-Operand Group

Instruction	Major States F S D E	Bus Cycles in Execute
MOV(B)	† * * †	DATO(B)††
CMP(B)	† * * †	NONE
BIT(B)	† * * †	NONE
BIC(B)	† * * †	DATO(B)††
BIS(B)	† * * †	DATO(B)††
ADD	† * * †	DATO††
SUB	† * * †	DATO††

†always used.
††modified DATO or DATOB (following DATIP).
*may or may not be used.

NOTE
Whenever destination address mode is 0, no DATO(B) bus cycle is performed.

Table B-3
Single-Operand Group

Instruction	Major States F S D E	Bus Cycles in Execute	Remarks
CLR(B)	† 0 * †	DATO(B)††	
COM(B)	† 0 * †	DATO(B)††	
INC(B)	† 0 * †	DATO(B)††	
DEC(B)	† 0 * †	DATO(B)††	
NEG(B)	† 0 * †	DATO(B)††	
ADC(B)	† 0 * †	DATO(B)††	
SBC(B)	† 0 * †	DATO(B)††	
TST(B)	† 0 * †	None	
ROR(B)	† 0 * †	DATO(B)††	
ROL(B)	† 0 * †	DATO(B)††	
ASR(B)	† 0 * †	DATO(B)††	
ASL(B)	† 0 * †	DATO(B)††	
JMP	† 0 * 0	None	
SWAB	† 0 † †	DATO(B)††	Goes direct to service major state

†always used.
††modified DATO or DATOB (following DATIP).
*may or may not be used
0 never used

NOTE
1. Whenever destination address mode is 0, no DATO(B) bus cycle is performed
2. In the case of the JMP instruction, the last bus operation in destination major state is not performed.

Table B-4
Operate Group

Instruction	Major States F S D E	Bus Cycles in Execute	Remarks
HALT	† 0 0 0	None	Transfers control to console.
WAIT	† 0 0 0	None	Processor stays in service major state until a bus request is received.
RESET	† 0 0 0	None	Transfers control to console.
RTI	† 0 0 †	DATI DATI	
IOT	† 0 0 0	None	See note below.
EMT	† 0 0 0	None	See note below.
TRAP	† 0 0 0	None	See note below.

†always used.
0 never used.

NOTE
For this instruction, bus cycles are performed during the service major state. Bus cycles for these instructions are given in Table B-8.

Table B-5
Subroutine Group

Instruction	Major States F S D E	Bus Cycles in Execute
JSR	† 0 † †	DATO
RTS	† 0 0 †	DATI

†always used.
0 never used.

NOTE
For the JSR instruction, the last bus operation in the destination major state is not performed.

Table B-6
Condition Code Operators

Instruction	Major States F S D E	Bus Cycles in Execute
CLC	† 0 0 †	None
CLV	† 0 0 †	None
CLZ	† 0 0 †	None
CLN	† 0 0 †	None
SEC	† 0 0 †	None
SEV	† 0 0 †	None
SEZ	† 0 0 †	None
SEN	† 0 0 †	None

†always used.
0 never used.

Table B-7
Conditional Branches

Instruction	Major States F S D E	Bus Cycles in Execute
BR	† 0 0 †	None
BNE	† 0 0 †	None
BEQ	† 0 0 †	None
BGE	† 0 0 †	None
BLT	† 0 0 †	None
BGT	† 0 0 †	None
BLE	† 0 0 †	None
BPL	† 0 0 †	None
BMI	† 0 0 †	None
BHI	† 0 0 †	None
BLOS	† 0 0 †	None
BVC	† 0 0 †	None
BVS	† 0 0 †	None
BCC	† 0 0 †	None
or		
BHIS	† 0 0 †	None
BCS	† 0 0 †	None
or		
BLO	† 0 0 †	None

†always used.
0 never used.

B.3.4 Service

In relation to the instruction set, the service major state has two primary functions: to implement certain asynchronous operations, such as console operations, and to process traps and interrupts.

For transfers to console control, such as those caused by HALT or RESET instructions, no bus cycles are performed. During trap and interrupt instructions, however, bus cycles occur during service. These cycles are listed in Table B-8.

Table B-8
Bus Cycles During Service

Instruction	Major States F S D E S	Bus Cycles in Service
IOT	† 0 0 0 †	DATO DATO DATI DATI
EMT	† 0 0 0 †	DATO DATO DATI DATI
TRAP	† 0 0 0 †	DATO DATO DATI DATI

†always used
0 never used.

APPENDIX C
UNIBUS PIN ASSIGNMENTS

Slots A1 and B1 (A4 and B4) are wired as shown in Tables C-1 and C-2.

Table C-1
Unibus Pin Assignments
(by pin numbers)

A1		B1	
Pin	**Signal**	**Pin**	**Signal**
AA1	INIT L	BA1	BG 6 H
AA2	Power (+5V)	BA2	Power (+5V)
AB1	INTR L	BB1	BG 5 H
AB2	Ground	BB2	Ground
AC1	D00 L	BC1	BR 5 L
AC2	Ground	BC2	Ground
AD1	D02 L	BD1	Ground
AD2	D01 L	BD2	BR 4 L
AE1	D04 L	BE1	Ground
AE2	D03 L	BE2	BG 4 H
AF1	D06 L	BF1	ACLO L
AF2	D05 L	BF2	DCLO L
AH1	D08 L	BH1	A01 L
AH2	D07 L	BH2	A00 L
AJ1	D10 L	BJ1	A03 L
AJ2	D09 L	BJ2	A02 L
AK1	D12 L	BK1	A05 L
AK2	D11 L	BK2	A04 L
AL1	D14 L	BL1	A07 L
AL2	D13 L	BL2	A06 L
AM1	PA L	BM1	A09 L
AM2	D15 L	BM2	A08 L
AN1	Ground	BN1	A11 L
AN2	PB L	BN2	A10 L
AP1	Ground	BP1	A13 L
AP2	BBSY L	BP2	A12 L
AR1	Ground	BR1	A15 L
AR2	SACK L	BR2	A14 L
AS1	Ground	BS1	A17 L
AS2	NPR L	BS2	A16 L
AT1	Ground	BT1	Ground
AT2	BR 7 L	BT2	C1 L

Table C-1 (Cont)
Unibus Pin Assignments
(by pin numbers)

A1		B1	
Pin	Signal	Pin	Signal
AU1	NPG H	BU1	SSYN L
AU2	BR 6 L	BU2	C0 L
AV1	BG 7 H	BV1	MSYN L
AV2	Ground	BV2	Ground

Table C-2
Unibus Pin Assignments
(by Signal name)

Signal	Pin	Signal	Pin
A00 L	BH2	D06 L	AF1
A01 L	BH1	D07 L	AH2
A02 L	BJ2	D08 L	AH1
A03 L	BJ1	D09 L	AJ2
A04 L	BK2	D10 L	AJ1
A05 L	BK1	D11 L	AK2
A06 L	BL2	D12 L	AK1
A07 L	BL1	D13 L	AL2
A08 L	BM2	D14 L	AL1
A09 L	BM1	D15 L	AM2
A10 L	BN2	Ground	AB2
A11 L	BN1	Ground	AC2
A12 L	BP2	Ground	AN1
A13 L	BP1	Ground	AP1
A14 L	BR2	Ground	AR1
A15 L	BR1	Ground	AS1
A16 L	BS2	Ground	AT1
A17 L	BS1	Ground	AV2
ACLO L	BF1	Ground	BB2
BBSY L	AP2	Ground	BC2
BG4 H	BE2	Ground	BD1
BG5 H	BB1	Ground	BE1
BG6 H	BA1	Ground	BT1
BG7 H	AV1	Ground	BV2
BR4 L	BD2	INIT L	AA1
BR5 L	BC1	INTR L	AB1
BR6 L	AU2	MSYN L	BV1
BR7 L	AT2	NPG H	AU1
C0 L	BU2	NPR L	AS2
C1 L	BT2	PA L	AM1
D00 L	AC1	PB L	AN2
D01 L	AD2	+5V*	AA2
D02 L	AD1	+5V*	BA2
D03 L	AE2	SACK L	AR2
D04 L	AE1	DCLO L	BF2
D05 L	AF2	SSYN L	BU1

*+5V is wired to these pins to supply power to the bus terminator only. +5V should never be connected via the Unibus between system units

Table C-3

BB11 Power Pin Assignments

Pin	Power
A1	-15V
A2	+ 5V
B1	-15V
B2	-15V
C1	-15V
C2	GND
D1	-15V
D2	GND
E1	-15V
E2	GND
F1	-15V
F2	GND
H1	-15V
H2	+ 5V
J1	-15V
J2	+ 5V
K1	-15V
K2	+ 5V
L1	-15V
L2	+ 5V
M1	-15V
M2	+ 5V
N1	GND
N2	-25V
P1	GND
P2	LTC L
R1	GND
R2	ACLO L
S1	GND
S2	DCLO L
T1	GND
T2	+ 8V
U1	GND
U2	+ 8V
V1	GND
V2	+ 8V

NOTE

Power is in module slot A3 of all system units mounted in BA11 mounting boxes equipped with H720 power supplies.

READER'S COMMENTS

Digital Equipment Corporation maintains a continuous effort to improve the quality and usefulness of its publications. To do this effectively we need user feedback – your critical evaluation of this manual.

Please comment on this manual's completeness, accuracy, organization, usability, and readability.

_____

_____

_____

_____

_____

Did you find errors in this manual? _____

_____

_____

_____

_____

How can this manual be improved? _____

_____

_____

_____

_____

_____

DEC also strives to keep its customers informed of current DEC software and publications. Thus, the following periodically distributed publications are available upon request. Please check the appropriate boxes for a current issue of the publication(s) desired.

☐ **Software Manual Update**, a quarterly collection of revisions to current software manuals.

☐ **User's Bookshelf**, a bibliography of current software manuals.

☐ **Program Library Price List**, a list of currently available software programs and manuals.

Please describe your position. _____

Name _____ Organization _____

Street _____ Department _____

City _____ State _____ Zip or Country _____

.....Fold Here .....

.....Do Not Tear – Fold Here and Staple.....

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

digital

Digital Equipment Corporation
Technical Documentation Department
146 Main Street (A&M Building)
Maynard, Massachusetts 01754

digital equipment corporation